Universität Stuttgart
Institut für Automatisierungs- und Softwaretechnik
Prof. Dr.-Ing. P. Göhner

01.03.1999

Diplomarbeit

M. Javad Irani **1674 DA**

Realisierung eines X-Windows Clients unter Java auf einem Network Computer

Betreuer: **Prof. Dr.-Ing. Peter Göhner**
Dipl.-Ing. Nasser Jazdi

Kurzfassung

Für die zukünftige Vernetzung von heterogenen Systemen in der Industrie steht mit den auf Network Computern (NCs) basierenden Thin Clients eine geeignete Plattform zur Verfügung. Um die Vorteile der Programmiersprache Java an NC zu nutzen, wurde das Betriebssystem JavaOS an NC angepaßt. Somit können die bisherigen Programme unter Windows, Unix und anderen Systemen auf dezentralen Network-Clients bedient werden.

Im Rahmen dieser Arbeit wurde zunächst ein auf PowerPC-basierender Network Computer Referenz Plattform (NCRP) der Firma IBM und das darauf laufende Betriebssystem JavaOS untersucht. Anschließend erfolgten Untersuchung und Konfiguration der Network Station (NS) mit Typ 8362-A53 der Firma IBM. Für den Zugriff auf eine serielle Schnittstelle wurden zwei kleine Programme geschrieben.

Anschließend wurde am Kaffeeautomaten etemat ECF 705 der Firma WMF eine externe Schnittstelle angebracht, damit man sie über NC via Netz steuern kann.

Abstract

The Thin Clients based on Network Computers (NCs) form an appropriate platform for any future networking of heterogenous systems in the industry.

To take advantage of the programming language Java in case of NC, the operating system JavaOS has been adapted to NC. That allows the operateion of all present programmers under Windows, Unix and another systems on decentral network clients.

One task of this work was the examination of a power PC-based network computer reference platform (NCRP) of IBM and the running operating system JavaOS. Later on, the Network Station (NS), type 8362-A53 of IBM has been examined and configurated, In order to allow the access to a serial interface two little programmes have been written.

After that an external interface has been fixed to the coffee automat etamat ECF 705 of WMF to allow the regulation of it by NC via net.

Inhaltsverzeichnis

1 Einführung 1
1.1 Einleitung 1
1.2 Aufgabenstellung 1
1.3 Aufbau der Arbeit 2
2 Konzepte der Programmiersprache Java 3
2.1 Java 3
2.2 Java Virtuelle Maschine (JVM) 5
2.3 JavaOS 7
2.3.1 Ein Kernel für Java 8
2.3.2 Device Drivers 8
2.3.3 Netzwerkprotokolle 9
2.3.4 Fensterdarstellung und Graphiken 9
2.3.5 HotJava und HotJava Views als das JavaOS-Desktop 9
2.3.6 Performance 9
2.3.7 Zusammenfassung 10
2.4 Zugriff auf Hardware unter Java 10
2.4.1 Java Communications API-ea3 11
3 Das Konzept der Client-Server Struktur 16
3.1 Network Computer Client-Server Architektur 16
3.2 Network Computer in der Realität 17
3.2.1 Die konventionelle Network Computer 17
3.2.2 Die Windows-Network Computer 17
3.2.3 Die Marketing Network Computer 18
3.3 IBM Network Computer 18
4 Network Computer Referenz Plattform 19
4.1 Network Computer Referenz Plattform 19
4.2 Boot-Methoden 21
4.3 Erforderliche Software auf dem Server 21
4.3.1 Konfiguration der DHCP- und TFTP-Server 22
4.4 Booten des NCs 24
4.4.1 Inbetriebnahme des NCs 25
4.4.2 JavaOS Properties 27
4.4.3 Laden der Applikationen in JavaOS 27
4.5 JavaOS und Java Communications API 28
4.6 Hardwarezugriff unter Java und JavaOS auf dem NC 28
5 Network Station, OS-9000 und JVM 31
5.1.1 Inbetriebnahme 32
5.1.2 Booten der Maschine 33

5.1.3 Erforderliche Informationen zu TCP/IP-Netzen ... 36
5.1.4 MAC-Adressen ... 37
5.1.5 Boot-Methoden ... 37
5.1.6 DHCP ... 37
5.2 IBM Network Station Manager ... 38
5.2.1 Voraussetzung und Installation ... 39
5.2.2 Voreinstellung der Boot- und Konfigurationsparameter ... 41
5.2.3 Arbeitsumgebung auf der NS ... 43
5.3 Untersuchung der Java Virtuelle Maschine (JVM) ... 44
5.4 Problematik und Fehlerbehebung ... 45
5.4.1 Softwareproblem ... 46
5.4.2 Hardwareproblem ... 46
6 Sensorik und Aktorik in einem Fließprozeß ... 48
6.1 Technische Realisierung ... 48
6.2 Temperaturmessung ... 49
6.2.1 Erfassung der Temperatur ... 51
6.2.2 Die Temperaturverstärker ... 51
6.2.3 AD 595 CQ von Analog Devices ... 52
6.3 Das Puffern von Sensorsignalen ... 53
6.4 Der Aktorumschalter ... 53
7 Zusammenfassung ... 55
8 Abkürzungen ... 57
9 Glossar ... 59
10 Literatur ... 63
Bücher, Skripte und Zeitschriften ... 63
Online-Literaturen ... 64
11 Anhang ... 65

Abbildungsverzeichnis

Abbildung 2.1: Entwicklungs- und Laufzeitumgebung 4
Abbildung 2.2: Java auf einem Hauptbetriebssystem 5
Abbildung 2.3: Java-Plattform auf dem JavaOS 7
Abbildung 4.1: Thin Client [12] 19
Abbildung 4.2: Block Diagramm der NCRP [7] 20
Abbildung 4.3: Konfigurations-Skript; dhcpdefaults.properties 23
Abbildung 4.4: Auszug aus dem dhcpserver.propertiesAbbildung 23
Abbildung 4.5: Inhalt der Datei netcenter.properties 24
Abbildung 4.6: System Information des NCs 24
Abbildung 4.7: Sourcecode der JavaOSApps.html-Seite des Applaunchers 26
Abbildung 4.8: der auskommentierte Quellcode von SimpleRead.java 29
Abbildung 5.1: Installation der Teile auf der Vorder- und Rückseite 33
Abbildung 5.2: Reihenfolge beim Booten 35
Abbildung 5.3: Verbindung von zwei NSs über ein LAN mit dem Server 37
Abbildung 5.4: Hauptanzeige des Programms NSM 39
Abbildung 5.5: Ausgabe des Programms Sysproperties 45
Abbildung 6.1: geschlossenes Thermopaar 49
Abbildung 6.2: offenes Thermoelement 49
Abbildung 6.3: Driftfehler des AD 595 CQ 53

Tabellenverzeichnis

Tabelle 4.1: Vergleich der Firmware des NCs 25
Tabelle 5.1: Einträge der NS 32
Tabelle 5.2: Netzplan 40
Tabelle 5.3: Installierte Software auf dem NT-Server 41
Tabelle 5.4: Boot- und Konfigurationsparameter (F3) 42
Tabelle 5.5: Boot-Datei, Pfad und Protokolle für die NVRAM-Boot-Methode (F4) 42
Tabelle 5.6: Konfigurationsdatei für die NVRAM-Boot-Methode (F5) 43

1 Einführung

1.1 Einleitung

Stark gesunkene Kosten sowie die zunehmende Integrationsdichte von PCs und Industriecomputern haben dazu beigetragen, daß der Anteil der Rechner in der Industrie in den letzten Jahren deutlich zugenommen hat. Neben der Verbesserung der Produktion dringen die Rechner auch zunehmend in sicherheitskritische Bereiche vor. Sie ermöglichen die Realisierung von Funktionen in Software, die durch rein mechanische und hydraulische Systeme nicht oder nicht in demselben Maße erbracht werden können.

Das Gesamtsystem von Rechnern in der Industrie kann grob in drei Schichten unterteilt werden. In der obersten Schicht sind Server, die in den meisten Fällen auf NT basieren. Diese Server dienen verschiedenen PCs in der zweiten Schicht. Sie bestehen aus unterschiedlichen Formen: handelsüblichen Personal Computern (PCs), Kompakt-PCs (Industrie-PC) usw. Zu der untersten Schicht gehören die Steuerungseinheiten. Alle drei Ebene sind mittels verschiedener Bussysteme vernetzt. Die wichtigste und häufigste Gruppe ist die mittlere Ebene. Durch diese Reihe von Rechnern kann man seitens des Servers auf Datenbanken zugreifen, auf der anderen Seite auf Steuerungseinheiten. Damit kann man alle Prozesse überwachen und steuern.

Die meisten eingesetzten PCs sind von Microsoft und Intel abhängig. Es wird in der Industrie angestrebt, eine heterogene Umgebung zu schaffen, damit man bei der Auswahl von Hardware und Software größtmögliche Freiheiten hat. Aus wirtschaftlicher Sicht ist es nicht immer möglich oder sinnvoll, von bereits bestehender Technik einfach auf ein neues System umzusteigen. Deshalb wurde in Betracht gezogen, die in der mittleren Schicht eingesetzten PCs durch NCs zu ersetzen. Damit werden die Portabilität von Java und die Vorteile des NCs vollkommen ausgenutzt.

1.2 Aufgabenstellung

Im Rahmen dieser Arbeit sollten als erstes die gängigen Methoden zur Referenzierung von Prozeßwerten in den Entwicklungssystemen der Steuerungshersteller untersucht werden. Diese basieren auf Datenbanken, die den Prozeßwerten symbolische Namen und Strukturen zuordnen. Hierbei sollten die Ansätze, die einen einfachen Zugriff auf die einzelnen Prozeßwerte auf Intranet-Ebene ermöglichen, verglichen und die geeignete Methode begründet werden. Anschließend sollte anhand einer prototypischen Implementierung deren Funktionalität überprüft werden. Da für Steuerungseinheiten die Schnittstellen des Clients sehr wichtig sind, sollte ein Interface unter Java erarbeitet und erstellt werden, damit man auf die Hardware der beiden Plattformen zugreifen kann.

Im Laufe der Zeit änderten sich die Prioritäten von der ursprünglichen Aufgabenstellung in dieser Arbeit dahingehend, daß dem Zugriff auf die Hardware höchste Priorität eingeräumt wurde. Aus diesem Grund beschäftigt sich die nun vorliegende Arbeit primär mit der Untersuchung der Plattformen.

Als Rechnerplattform dienen ein NCRP und eine weiterentwickelte Network Station (NS), die jeweils mit einem PowerPC-Prozessor ausgestattet sind. Auf dem NC läuft das auf Java basierende Betriebssystem JavaOS und auf der NS das Echtzeitbetriebssystem OS-9000, auf dem wiederum eine Java Virtuelle Maschine laufen kann.

1.3 Aufbau der Arbeit

Die vorliegende Arbeit ist in sechs Kapitel gegliedert. In den ersten drei Kapiteln werden die Grundlagen dieser Arbeit, d.h. die Philosophie der Programmiersprache Java und des JavaOS und die Client/Server-Struktur dargestellt. In den Kapiteln 4 und 5 wird der Hauptteil der Arbeit beschrieben, die Untersuchung der Plattformen und die Erstellung der Konzeption. Die Ergebnisse der Untersuchungen, Beispielprogramme und Probleme werden ebenfalls in Kapitel 4 und 5 behandelt. Kapitel 6 schließt diese Arbeit mit einer Zusammenfassung und einem Ausblick ab.

2 Konzepte der Programmiersprache Java

Als SUN die Programmiersprache JAVA im Jahr 1995 vorstellte, war sie ursprünglich dafür gedacht, die Steuerung von Klein- und Haushaltsgeräten zu übernehmen. Heute ist JAVA eine allgemein anerkannte und weit verbreitete Programmiersprache.

2.1 Java

Die Java-Plattform ist eine neue Software-Plattform für die Zustellung und Ausführung hoher, interaktiver, dynamischer und sicherer Appletts und Anwendungen unter anderem auch auf NC. Die Java-Plattform sitzt auf vorhandenen Plattformen und führt Bytecodes durch, die nicht zu physikalischen Maschinen spezifisch sind, sondern sie sind maschinelle Anweisungen für eine virtuelle Maschine.

Java ist eine Programmiersprache, mit der die Unterschiede zwischen verschiedenen Plattformen überbrückt werden können. Javas Zielsetzung „einmal schreiben, überall ausführen“ bezieht sich auf ihre Tragfähigkeit und auf die Möglichkeit, ein Java-Programm auf verschiedenen Plattformen auszuführen. Zur Ausführung von Java-Anwendungen wird ein Paket mit Programmen zur Java-Aktivierung benötigt, das als „Java Virtual Machine” (JVM) bezeichnet wird. Die Verwendung von JVM auf Clients ohne Plattenspeicher, wie z.B. NC, ermöglicht dem Benutzer den Zugriff auf Anwendungen, ohne einen permanenten Plattenspeicher auf dem NC oder auf dem Server verwenden zu müssen. Er kann eine JVM von Servern herunterladen und anschließend Java-Programme starten und konfigurieren. Es gibt zwei Arten von Java-Programmen:

- Applets — erfordern einen Browser oder einen Applet Viewer
- Anwendungen — werden direkt angezeigt

Applets stützen sich auf einen Browser oder einen Applet Viewer, der das Fenster und die graphischen Darstellungen bereitstellt. Im allgemeinen reagieren Browser restriktiv auf Applets, da sie über das Internet heruntergeladen werden. Anders ausgedrückt kann der Browser Applets daran hindern, lokale Dateien zu lesen und Dateien lokal zu speichern sowie Verbindungen zu anderen Maschinen herzustellen als den Maschinen, von denen sie heruntergeladen wurden. Diese Einschränkungen schützen Benutzer vor virusinfizierten Programmen und gewährleisten eine sichere Umgebung zur Überprüfung von Programmen im Internet. An dem NC können Java-Applets und -Anwendungen ausgeführt werden.

Es kann jeweils nur eine Java-Anwendung am NC oder an der NS ausgeführt werden. Wenn eine Java-Anwendung läuft, können keine Applets auf der Arbeitsoberfläche oder im Browser ausgeführt werden.

Dafür ist Java aber eine multithreadfähige Sprache, d.h. sie unterstützt mehrere sogenannte Threads. Das sind kleine eigenständige Prozesse, die verschiedene Aufgaben erledigen können. Ein wichtiger Vorteil der Multithread-Fähigkeit ist, daß auf diese Weise das interaktive Verhalten graphischer Anwendungen gegenüber dem Benutzer verbessert wird.

Eine Java-Entwicklungsumgebung umfaßt die Compilierzeit- und Laufzeitumgebung, wie in Abbildung 2.1 gezeigt ist. Die Java-Plattform wird durch die Laufzeitumgebung dargestellt. Der Entwickler schreibt die Java-Quellcodes (Java-Dateien) und compiliert sie zu den Bytecodes (Classdateien). Diese Bytecodes sind Anweisungen für die JVM.

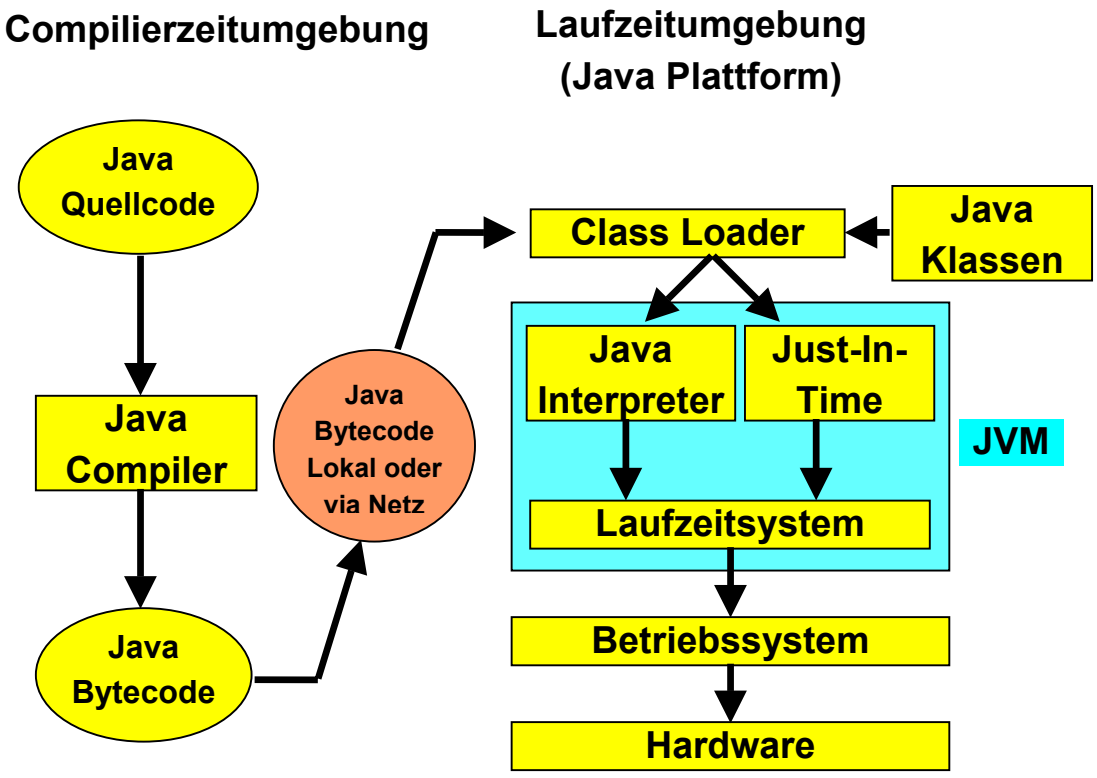


Abbildung 2.1: Entwicklungs- und Laufzeitumgebung

Während jede zugrundeliegende Plattform ihre eigene Implementierung der Java Virtuellen Maschine hat, gibt es nur eine Virtuelle Maschinen Spezifikation. Aufgrund dieses Umstandes kann die Java-Plattform eine Standard- und konstante Programmierungsschnittstelle zu den Applets und den Anwendungen auf jeder möglichen Hardware zur Verfügung stellen. Folglich ist die Java-Plattform für das Internet ideal, in dem ein Programm auf jedem möglichen Computer in der Welt lauffähig sein sollte. Das Java-API liefert die Spezifikation, die vom Programmierer für eine Anwendung oder ein Applet verwendet werden kann. Das Java Development Kit (JDK) definiert die Java-API, die sowohl Applikationen als auch Browsers und Browserapplets unterstützt. Die Java-API ist in allen Java-Laufzeiten dieselbe, einschließlich JavaOS, MS-Windows, UNIX und aller Macintosh Implementierungen.

2.2 Java Virtuelle Maschine (JVM)

Wenn die Java-Plattform mit einem Betriebssystem des Hauptrechners benutzt wird, können die Virtuellen Maschinen und die Java Foundationklassen entweder innerhalb des Betriebssystems oder innerhalb einer Anwendung wie ein Web-Browser eingebettet werden. Um die Java-Plattform zu unterstützen, gibt das Betriebssystem des Hauptrechners der Java-Plattform etwas Unterstützung. Jede der Haupteigenschaften der Java-Plattform, direkt oder indirekt, projiziert Anforderungen auf ein Betriebssystem des Hauptrechners. Viele der Sprach- und Hilfsklassen angenommener Betriebssysteme sind vorhanden.

Abbildung 2.2 veranschaulicht die Software-Architektur, wenn die Java-Plattform auf einem Betriebssystem des Hauptrechners verwendet wird. In Abbildung 2.2 ist alles, was über dem Java-API liegt, plattformunabhängig. Die Architektur ist auf allen Plattformen dieselbe.

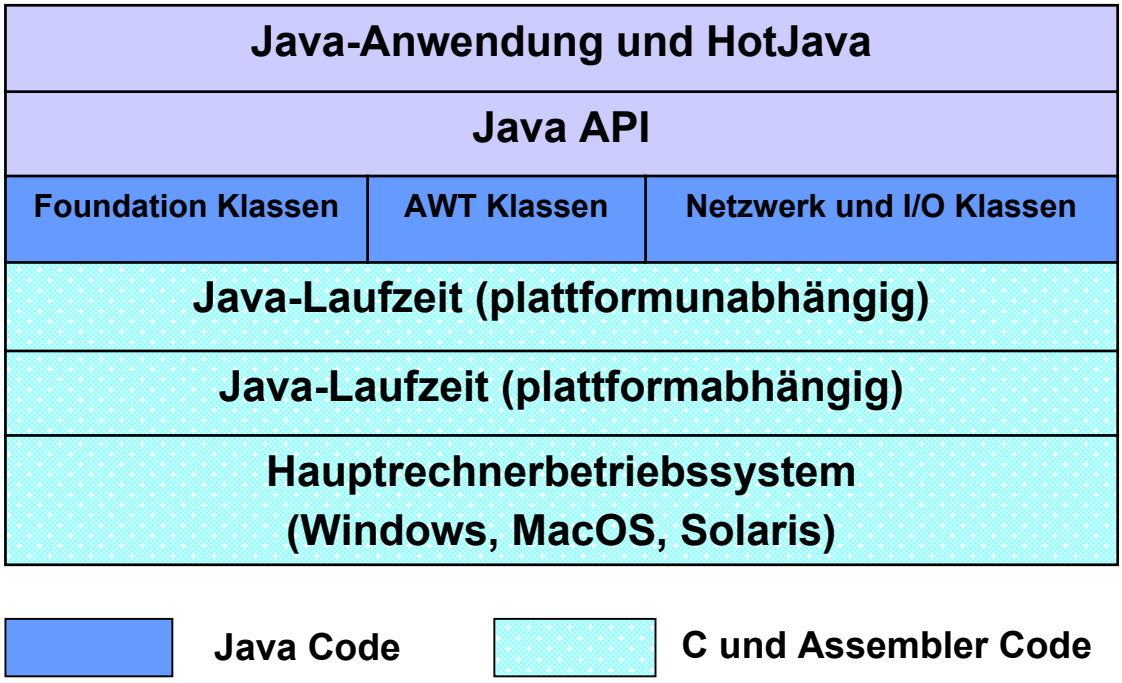


Abbildung 2.2: Java auf einem Hauptbetriebssystem

Die Java-API wird durch Klassen, einschließlich der Sprach- und Hilfsklassen, der AWT, des Netzes und I/O Klassen, geliefert. Alle Komponenten sind in Java geschrieben. Die Java-Laufzeit, die den Interpreter und den Abfallsammler (garbage) einschließt, ist überwiegend in dem plattformunabhängigen C-Code geschrieben. Wenn die Java-Laufzeit auf eine neue Plattform übertragen wird, muß ein spezifischer plattformabhängiger Code verwendet werden. Diese Java-Laufzeit ist (abhängig von der anschließenden Schnittstelle der Plattform) in Abbildung 2.2 erläutert. Das Betriebssystem des Hauptrechners muß folgendes zur Verfügung stellen:

- Multithread-Unterstützung für die Java-Laufzeit; zumindest die ursprüngliche Unterstützung für eine Kontext-Schaltung. Wenn das Hauptrechnersystem bessere Unterstützung für Threads bietet, kann diese Unterstützung benutzt werden.
- Speicherallozierung. Obwohl die Java-Laufzeit ihren eigenen Heap verwaltet und Garbage (Abfallansammlung) umfaßt, benötigt sie noch einen Mechanismus für das Zuordnen des Speichers, den sie verwaltet.
- Fensterdarstellung und Graphikunterstützung für den AWT, der die graphische Benutzungsschnittstelle liefert.
- Die Standardnetzwerkprotokolle zum Unterstützen der Java-Vernetzungsklassen.

Wenn die Java-Plattform portiert ist, muß der Entwickler folgendes tun:

- Abbilden des AWT auf Windows und auf dem Graphik-Teilsystem, das durch das Hauptrechnersystem bereitgestellt wird.
- Abbilden der Vernetzungsklassen auf nativen Vernetzungscode des Systems, das z.B. unterschiedliche Systemaufrufe für Operationen auf Sockets haben kann.
- Abbilden der filebezogenen I/O Klassen zum Hauptrechnerfilesystem, das eine andere Bezeichnung der Filenamen verwenden könnte.
- Portierung des plattformabhängigen Teils der JVM zu den bestimmten Systemaufrufen für Speicherallozierung und Threadsverwaltung.

2.3 JavaOS

JavaOS liefert eine unabhängige Java-Umgebung, das heißt., die entwickelten Anwendungen für die Java-Plattform, die JavaOS verwendet, können ohne die Unterstützung eines zusätzlichen Betriebssystems laufen. Die Anwendungen, die zum Ablauf auf einer Maschine ohne ein Hostbetriebssystem geschrieben sind, könnten auf einer Maschine mit Hostbetriebssystem laufen. JavaOS unterstützt folgende Merkmale, um eine Java-Plattform zu realisieren:

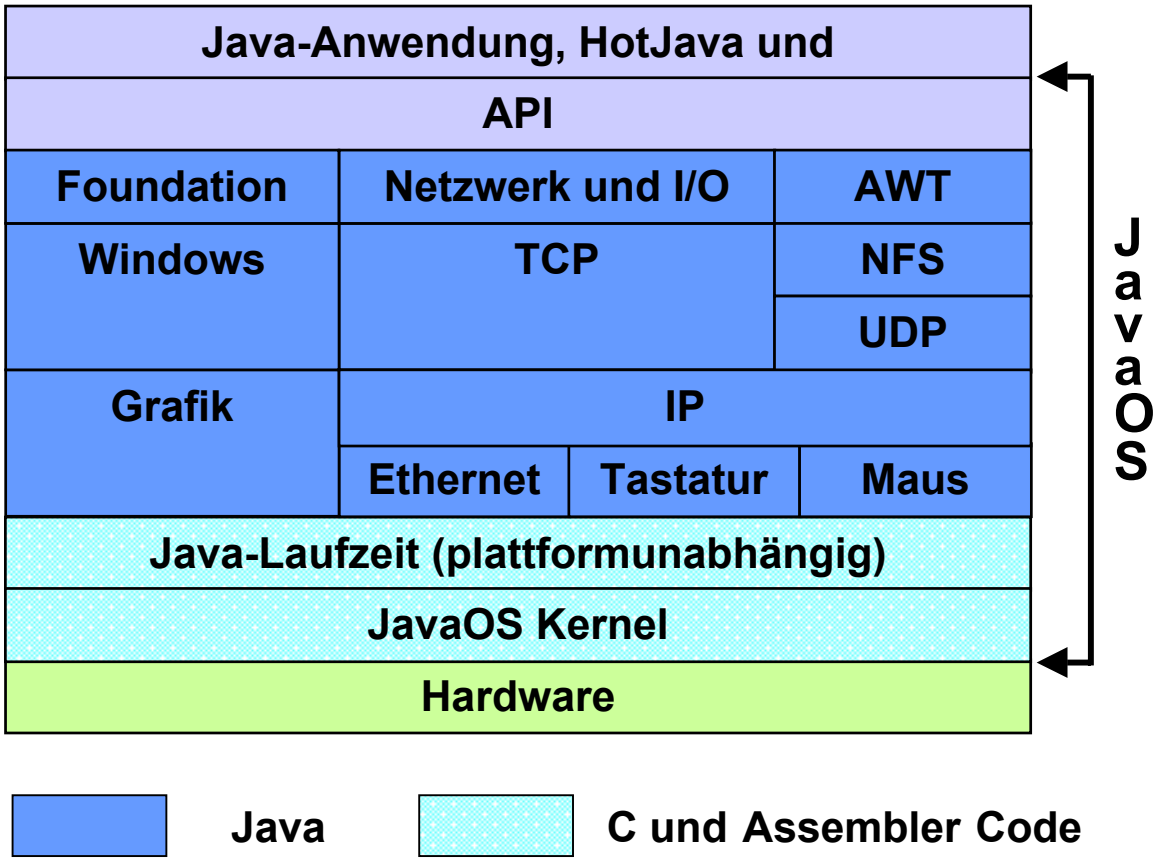


Abbildung 2.3: Java-Plattform auf dem JavaOS

- Unterstützung der JVM mit einem Kernel für Java
- Unterstützung der AWT, vernetzungs- und filebezogene I/O-Klassen
- Treiber für die Steuerung einer Bildschirmanzeige, der Netzschnittstelle, der Maus und der Tastatur
- Unterstützung der vollen Java-API

Abbildung 2.3 zeigt die Software-Architektur, die bei einer Java-Plattform ohne ein Betriebssystem des Hauptrechners verwendet wird. Wie Abbildung 2.2zeigt sind Programme über dem Java-API plattformunabhängige Java-Anwendungenen und -Applets.

2.3.1 Ein Kernel für Java

Innerhalb JavaOS führt die Java-Laufzeit die User-level Anwendungen, die System-level Windows, die Graphiken, die Vernetzung und den Treibercode durch. Die niedrigste Schicht des Codes bearbeitet die Aufgaben, die häufig in Mikro- oder in den Nanokerneln gefunden werden. Der Kernel für Java enthält die niedrigen Funktionen, die durch die JVM angefordert werden. Diese angeforderte Funktionalität fällt in die folgenden Kategorien:

Laden des Betriebssystems, Threads, Speicherverwaltung, Monitor, Filesystem, Timing, Verwaltung der nativen Codebibliothek, Interrupt, DMA (Direct Memory Access), Debugging, diverse Steuerung der Plattform.

Bootsysteme für Netz, ROM (Read Only Memory), RAM (Random Access Memory), CD-ROM (Compact Disk - Read Only Memory), Floppy und Festplatten sind alle möglich. Solange der Bootstrap läuft, ordnet er einige Speicherbereiche für z.B. den Javaheap, und verschiedene I/O-Deviceregister zu.

Der Threadsunterstützungscode aktiviert die JVM, um Zusammenhänge zwischen den vielen Threads zu schalten, die gewöhnlich im System laufen. Wegen den Sicherheitskonzepten, die durch die Programmiersprache Java bereitgestellt wird, können JavaOS und alle Anwendungen in einem einzelnen Adressenbereich laufen. Dieses vereinfacht und optimiert den Kontext-Schaltungs-Code in JavaOS.

JavaOS benötigt keine Speicherverwaltungseinheit (Memory Management Unit, MMU), sondern es kann ein MMU verwendet werden, um mehrere zerlegte Strecken an die physikalischen Speicher angrenzend zu erstellen. Die Programmiersprache Java beseitigt die direkte Handhabung von Speichern durch Einkapseln aller Zugriffe in Objekte, Klassen und durch die Ansammlung automatisch freier Speicher.
Anders als viele Betriebssysteme, die heute in Gebrauch sind, funktioniert JavaOS nicht unter der Annahme der mehrfachen virtuellen Adresse. Statt dessen funktioniert JavaOS in einem einzelnen virtuellen Adressenbereich.

2.3.2 Device Drivers

Alle Gerätetreiber in JavaOS sind in der Programmiersprache Java geschrieben. Dies ist für Portabilität wichtig.

Es gibt sicher Dinge, die beinahe jeder Treiber in reinem Java-Quellcode nicht ausführen kann. Diese sind in zwei kleinen unterstützten Klassen in C geschrieben. Die `Memory` Klasse aktiviert den Treiber zum Zugriff und zum Ändern spezifischer Bytes und Worte der Speicherung. Die `Interrupt` Klasse verwaltet das Verschicken von Interrupthandle. Die Methoden dieser Klassen sind nicht für irgendwelche Java-Anwendungen zugänglich.

2.3.3 Netzwerkprotokolle

JavaOS schließt eine große Folge der Netzwerkprotokolle, die alle in der Programmiersprache Java geschrieben sind, ein. Diese Protokolle umfassen die grundlegenden Transport- und Routingsmechanismen, die durch die TCP-, UDP-, IP- und ICMP-Standards spezifiziert sind. JavaOS verwendet DNS und NIS (Network Information Service), um Hostname, Benutzername und -kennwörter während des LOGON zu überprüfen.

JavaOS unterstützt beide ARP (Address Resolution Protocol) und DHCP, um die Netzwerkadressen eines Gerätes zu erkennen. Damit läßt sich aber keine Administration für Maschinen installieren.

Eine Maschine, auf der JavaOS läuft, kann auf die Dateien als Client eines NFS-Servers zugreifen. Sie kann mit dem SNMP verwaltet werden. Die JavaOS-Maschinen können die Zeit von einem Netzserver erhalten, welcher auch die Installation und Administration vereinfacht.

2.3.4 Fensterdarstellung und Graphiken

Außer den Netzwerkprotokollen sind die Fensterdarstellung und das Graphik-Teilsystem das größte Stück des Betriebssystems. JavaOS benutzt die kleine AWT-Bibliothek, um in Java gegründete Implementierungen von Widgets wie Tasten, Menüs und Scrollbars zur Verfügung zu stellen.

2.3.5 HotJava und HotJava Views als das JavaOS-Desktop

HotJava ist die erste Desktop Benutzungsschnittstelle, die in der Programmiersprache Java geschrieben ist. Wenn es mit HotJava oder HotJava Viewer kombiniert wird, kann JavaOS als komplettes Multitasking mit graphischer Betriebsumgebung arbeiten. HotJava kann als Manager der Desktop Metapher und der mehrfachen Fensterdarstellung dienen, von denen jedes zum Blättern von HTML-Seiten und zum Start von einem oder mehreren Java-Applets befähigt ist. Hot-Javas Einfachheit und Erweiterbarkeit bildet es zu einem idealen Rahmen für JavaOS-Anwendungen. JavaOS kann andere Hauptprogramme, außer HotJava und HotJavaViews, auch laufen lassen.

2.3.6 Performance

Dieser Abschnitt behandelt einige Leistungsvorteile der Anwendung der Java-Plattform, die auf JavaOS läuft.

2.3.6.1 Geschwindigkeit

Zur Zeit ist JavaOS auf minimale Leistung abgestimmt. Der JIT (Just In Time) Compiler zum Übersetzen des Bytecodes in einen Maschinencode wird nicht benutzt. Er macht kaum Gebrauch von der nativen Methode. Aber die Messungen haben gezeigt, daß seine Leistung gut und sogar in manchen Bereichen besser als C oder C++ ist. Die TCP/IP Leistung auf JavaOS ist bereits doppelt so groß wie das ursprünglich festgelegte Ziel und ist damit für den Web-Browser mehr als ausreichend.

2.3.6.2 Speicherbedarf

Ein komplettes System könnte mit einer Gesamtmenge von 4MB des ROM und 4MB des RAM aufgebaut werden. Im ROM befindet sich der ganze Code für JavaOS selbst, einschließlich des Kerncodes, der Treiber, JVM und Standardklassen, der JavaOS-Windows, Graphiken und Vernetzungsbestandteile, des Codes für HotJava. Das ROM könnte mit 1MB der Bitmap für die verschiedenen Schriftarten, wie serif, sans-serif, Schreibmaschine, mehreren Größen und Schreibweisen wie Fettdruck und Italic auskommen.

2.3.7 Zusammenfassung

Zusammengefaßt läßt sich sagen, daß JavaOS eine neue Software-Plattform ist, die Java-Anwendungen direkt auf der Hardware laufen läßt, ohne ein Betriebssystem des Hauptrechners zu benötigen.

JavaOS umfaßt die JVM, die Standardpakete der Klassen und gerade genügend OS-Code, um sie zu unterstützen. Der OS-Code umfaßt den niedrigen Code, geschrieben in C- oder Assemblersprache, plus Gerätetreiber, Vernetzung, Fensterdarstellung (Windowing) und einen Graphikwiedergabecode (größtenteils in Java geschrieben).

JavaOS beseitigt die Kompliziertheit der Betriebssysteme und Unkosten des Hauptrechners; aktiviert neue einfache Klassen, intelligente und dynamische Netzwerkeinheiten, die niedrigere Kosten verursachen. JavaOS wurde gezielt für Systeme wie Internet-Computer, Intranet-Terminals für Unternehmen und eingebettete Geräte entwickelt.

2.4 Zugriff auf Hardware unter Java

In der Regel ermöglicht die Programmiersprache Java keinen direkten Zugriff auf physikalische Geräte.

Sie ist eine virtuelle Maschine und verhält sich wie eine Abstraktionsebene. Es kann sein, daß die I/O Geräte schon unter einer Hochprogrammiersprache geschrieben sind. Aber

Java bietet zwei Möglichkeiten, um auf die Hardware zuzugreifen: Das Java Native Interface (JNI) zur Implementierung für gerätespezifisches Verhalten und die Java Communications API. Die Native Codes unterstützen die serielle Schnittstelle auf verschiedenen Plattformen [4]. Als Beispiel kann man sich in PortIO-Packet von Central-Data einarbeiten [4],[14]. Dieses Paket enthält eine "high-Level" Klasse für einen Portdriver und einen Port. Die zusätzliche Klasse PortPrivate kann für gerätespezifische Verhalten erweitert werden.

2.4.1 Java Communications API-ea3

Die Java Communications API, die als Standard-Erweiterung für die Java Plattform gilt, kann verwendet werden, um plattformunabhängige Kommunikationsanwendungen für Technologien wie Sprachpost (voice mail), Telefax und Smart Cards zu schreiben. Diese Communications API enthält einige Beispielanwendungen, mit denen sich die Funktionalität der Ports überprüfen läßt. Diese Version der Communications API enthält Unterstützung für serielle Schnittstellen RS232 und IEEE 1284 parallele Ports. Mit aktualisierter Funktionalität kann folgendes gewährleistet werden:

1. Aufzählung der vorhandenen Ports auf dem System
2. Öffnen und Anspruch auf Besitz der Ports
3. Behebung des Portbesitzes zwischen mehrfachen Anwendungen
4. Durchführung der asynchronen und synchronen I/O auf Ports
5. Empfangen der Bean-Style-Ereignisse und die Zustandsänderungen des Communication-Port. Es muß beachtet werden, daß die gesamte Software nur auf den Systemen mit JDK/JRE 1.1.x (für JavaOS JDK1.1.4) ausgeführt werden kann. Diese unterstützen nur Java-Anwendungen und keine Applets.

Die Communications API Dokumentation kann nicht als Tutorial dienen. Der Programmierer muß Programmierungsgrundlagen für Serial- und Parallelport beherrschen, damit er die Communications API verwenden kann.

Es gibt drei Klassenstufen in der Java Communikations Extension API:

- Hochstufenklassen (high-level) wie `CommPortIdentifier` und `CommPort`, die den Zugriff und Besitz der Kommunikationsports verwalten.
- Niedrigstufenklassen (low-level) wie `SerialPort` und `ParallelPort`, die eine Schnittstelle zu den physikalischen Kommunikationsports zur Verfügung stellen. Diese aktuelle Freigabe der Java Kommunikation Extension API ermöglicht Zugriffe auf die seriellen- (RS-232) und parallelen (IEEE 1284) Ports.
- Treiberstufenklassen (Driver-level), die eine Schnittstelle zwischen den niedrigen Klassen und dem zugrundeliegenden Betriebssystem liefern. Treiberstufenklassen sind ein Teil der Implementierung, aber nicht Teil der Java Communications Extension API. Sie sollten nicht vom Programmierer in Anwendungen benutzt werden.

2.4.1.1 `Javax.comm` Erweiterungspaket

Das `javax.comm`-Paket liefert die folgenden Grunddienste:

- Aufzählung der vorhandenen Ports auf dem System. Die statische Methode `CommPortIdentifier.getPortIdentifiers` bringt ein Aufzählungsobjekt zurück, das ein `CommPortIdentifier` Objekt für jeden vorhandenen Port enthält. Dieses `CommPortIdentifier` Objekt ist die zentrale Einheit für die Zugriffsteuerung zu einem Kommunikationsport.
- Öffnen und Besitzanspruch der Kommunikationsports, indem man die Hochstufenmethoden in seinem `CommPortIdentifier` Objekt verwendet.
- Beheben der Portbesitzabsicht zwischen mehrfachen Java-Anwendungen. Ereignisse werden fortgepflanzt, um interessante Anwendungen zu benachrichtigen und dem Inhaber des Ports zu erlauben, Besitz abzutreten. `PortInUseException` wird benutzt, wenn eine Anwendung den Port nicht öffnen kann.
- Durchführung von asynchronem und synchronem I/O auf dem Kommunikationsport. Niedere Klassen wie `SerialPort` und `ParallelPort` haben Methoden für die Verwaltung vom I/O auf dem Kommunikationsport.
- Empfangen von Ereignissen, die den Kommunikationsport Zustandsänderungen beschreiben, z.B. wenn eine serielle Schnittstelle eine Zustandsänderung für Carrier Detect, Ring Indicator, DTR, etc. hat, ermittelt `SerialPort` Objekt ein `SerialPortEvent`, das die Zustandsänderung beschreibt.

2.4.1.2 Klassenhierarchie

Nachfolgend ist die Hierarchie der Klassen und Schnittstellen, wie sie in der Communications API definiert ist, noch einmal beschrieben:

Klasse java.lang.Object
- Schnittstelle javax.comm.`CommDriver`
- Klasse javax.comm.`CommPort`
  - Klasse javax.comm.`ParallelPort`
  - Klasse javax.comm.`SerialPort`
- Klasse javax.comm.`CommPortIdentifier`
- Schnittstelle javax.comm.`CommPortOwnershipListener` (Subklasse java.util.`EventListener`)
- Klasse java.util.`EventObject` (implements java.io.Serializable)
  - Klasse javax.comm.`ParallelPortEvent`
  - Klasse javax.comm.`SerialPortEvent`
- Schnittstelle javax.comm.`ParallelPortEventListener` (extends java.util.`EventListener`)
- Schnittstelle javax.comm.`SerialPortEventListener` (Subklasse java.util.`EventListener`)
- Klasse java.lang.`Throwable` (implementiert java.io.Serializable)
  - Klasse java.lang.`Exception`
    - Klasse javax.comm.`NoSuchPortException`
    - Klasse javax.comm.`PortInUseException`
    - Klasse javax.comm.`UnsupportedCommOperationException`

Die wichtigen Komponenten dieses Pakets werden unten noch einmal aufgelistet:

- `CommDriver`
- `CommPortOwnershipListener`
- `ParallelPortEventListener`
- `SerialPortEventListener`

- `CommPort`
- `CommPortIdentifier`
- `ParallelPort`
- `ParallelPortEvent`
- `SerialPort`
- `SerialPortEvent`

- `NoSuchPortException`
- `PortInUseException`
- `UnsupportedCommOperationException`

Als Beispiel werden einige wichtige Bestandteile des Pakets näher beschrieben.

CommPortOwnershipListener

pflanzt verschiedene Kommunikationsportbesitzereignisse fort. Wenn ein Port geöffnet ist, wird ein `CommPortOwnership`-Ereignis der Type `PORT_OWNED` fortgepflanzt. Wenn ein Port geschlossen ist, wird ein `CommPortOwnership`-Ereignis der Type `PORT_UNOWNED` fortgepflanzt. Mehrfache Anwendungen, die den Besitz eines Kommunikationsportes suchen, können ihre Unterschiede wie folgt beheben:

- Aapp ruft `openPort` auf und nimmt den Besitz des Ports an
- Zapp ruft `openPort` einmal später auf
- Bei der Verarbeitung von Zapp's `openPort`, pflanzt `CommPortIdentifier` ein `CommPortOwnership` Ereignis mit der Ereignistype `PORT_OWNERSHIP_REQUESTED` fort
- Wenn ABCapp bereit ist, den Besitz abzugeben, ruft sie `closePort` innerhalb vom Ereignisrückruf auf
- Nachdem das Ereignis abgefeuert worden ist, überprüft es der `CommPortIdentifier`, um zu sehen, ob der Besitz abgegeben wurde, und wenn ja,

wird der Besitz des Ports zu XYZapp mittels Zurückbringen des Erfolges von `openPort` umgedreht.

Anmerkung: Wenn ein `closePort` innerhalb eines `CommPortOwnership` Ereignisrückrufs aufgerufen wird, wird ein neues `CommPortOwnership` Ereignis nicht festgelegt.

CommPort (Kommunikationsport)

`CommPort` ist eine abstrakte Klasse, die den Kommunikationsport beschreibt. Dieser Kommunikationsport wird durch das Betriebssystem zur Verfügung gestellt. Er umfaßt hochqualifizierte Methoden (high-level) für die Steuerung des I/Os. Diese Methoden sind für die unterschiedlichen Typen der Kommunikationsports gleich. `SerialPort` und `ParallelPort` sind Unterklassen von `CommPort`, die zusätzliche Methoden zur niedrigen Steuerung (low-level) der physikalischen Kommunikationsports umfassen.

Es gibt keinen allgemeinen Konstruktor für `CommPort`. Stattdessen sollte eine Anwendung die statische Methode `CommPortIdentifier.getPortIdentifiers` verwenden, um eine Liste der vorhandenen Ports festzulegen. Sie wählt dann einen Port von dieser Liste und ruft `CommPortIdentifier.openPort` auf, um ein `CommPort` Objekt zu erstellen. Schließlich wirft sie das `CommPort` Objekt zu einer physikalischen Kommunikationsklasse wie `SerialPort` oder `ParallelPort`.

Nachdem ein Kommunikationsport gekennzeichnet und geöffnet worden ist, kann er mit den Methoden in den niedrigstufigen Klassen, wie `SerialPort` und `ParallelPort`, konfiguriert werden. Dann kann ein I/O Stream zum Lesen und Schreiben der Daten geöffnet werden. Sobald die Anwendung mit dem Port fertig ist, muß sie die `closePort` Methode aufrufen. Danach darf die Anwendung keine Methoden in dem Port Objekt aufrufen. Dies verursacht, daß ein `java.lang.IllegalStateException` ausgeworfen wird.

CommPortIdentifier (Kommunikationsportverwalter)

`CommPortIdentifier` ist die zentrale Klasse für die Zugriffskontrolle zu den Kommunikationsports. Sie umfaßt Methoden für:

- Bestimmung der Kommunikationsports (durch den Treiber zur Verfügung gestellt),
- Öffnung der Kommunikationsports für I/O Operationen,
- Bestimmung des Portbesitzes,
- Behebung der Portbesitzabsicht,
- Verwaltungsereignisse, die Änderungen im Portbesitzstatus anzeigen.

Eine Anwendung verwendet zuerst Methoden im `CommPortIdentifier`, um mit dem Treiber zu verhandeln. Nachdem das Vorhandensein der Kommunikationsports entdeckt worden ist, wird ein Port zum Öffnen ausgewählt. Er verwendet dann die Methoden in anderen Klassen, wie `CommPort, ParallelPort` und `SerialPort`, um mit dem Port zu kommunizieren.

ParallelPort (paralleler Kommunikationsport)

`ParallelPort` beschreibt die niedrige Schnittstelle (low-level) zu einem parallelen Kommunikationsport, der durch das Betriebssystem zur Verfügung gestellt wird. `ParallelPort` definiert die minimal angeforderte Funktionalität für die parallelen Kommunikationsports.

SerialPort (RS-232 serieller Kommunikationsport)

SerialPort beschreibt die niedrige Schnittstelle (low-level) zu einem seriellen Kommunikationsport, der durch das Beriebssystem zur Verfügung gestellt wird. SerialPort definiert die minimal angeforderte Funktionalität für die seriellen Kommunikationsports.

3 Das Konzept der Client-Server Struktur

Die Client/Server-Technologie hat einen folgenreichen Paradigmenwechsel in der Industrie hervorgerufen. Sie ersetzt die monolitischen Großrechner-Applikationen durch solche, die in Client- und Server-Seiten aufgeteilt sind. Der Client (in der Regel ein PC oder NC) liefert die graphische Benutzeroberfläche, während der Server den Zugriff auf gemeinsame Ressourcen (typischerweise eine Datenbank) bietet. Verteilte Objekte und das Internet bilden wiederum einen Paradigmenwechsel innerhalb dieses Paradigmenwechsels: Sie stellen eine neue Client/Server-Revolution innerhalb der Client/Server-Revolution dar. Objekte teilen die Client- und Server-Seiten einer Applikation auf in aufeinander abgestimmte Komponenten, die zusammenarbeiten und dabei Netzwerke umspannen können. Das Java-Web ist die “Killer-Applikation”, die die Objekte der breiten Masse nahebringt. Die Kombination von verteilten Objekten und dem Web (das Objekt-Web) erfindet die Client/Server-Technologie aufs neue.

Da inzwischen für dieses Konzept viel geforscht und entwickelt worden ist, wurde die Realisierung einer Client/Server-Struktur aufgegeben. Man kann die vorhandenen Protokolle wie ICA (Independet Computing Architecture) verwenden, um solche Strukturen zu realisieren [14]. Im folgenden werden die zusätzlichen Varianten für Client/Server im Bereich NC vorgestellt und ihre Umgebungen miteinander verglichen.

3.1 Network Computer Client-Server Architektur

Wo bisher umfangreiche Windows-Anwendungen auf teuren PCs liefen, soll bald ein kleiner und kostengünstiger NC den Platz einnehmen. Seine vergleichsweise geringe Rechenleistung gleicht dieser aus durch auf das Wesentliche beschränkte Software und ein intelligentes Umfeld, denn Programme und selbst Daten werden nicht mehr lokal auf der Festplatte gespeichert, sondern auf einem Internet-Server. Somit fallen für den Benutzer keine Installations- oder Wartungsaufgaben an und man spart eine Menge Geld. Der NC kommt ohne Intel-Prozessor und ohne Microsoft-Software aus. Seine Daten speichert er auf einem Server oder Mainframe. Anwendungen für den NC sollen in Java geschrieben werden, was zusätzlich den Vorteil bietet, daß diese Anwendungen auch auf allen bereits installierten Plattformen lauffähig sind.

Für Unternehmen, welche ihr Geld mit Großrechensystemen, Mainframes, Serversoftware, usw. verdienen, bietet sich der NC daher als ideale Plattform an, in ihrem Bereich neue Märkte zu erschließen. Ohne Server verliert der NC natürlich seine Daseinsberechtigung. Dieser Server könnte auch bei einem ISP (Internet Service Provider) stehen. Für Heimanwender würde dies allerdings bedeuten, daß, eine dauernde Verbindung während der Arbeit zum ISP notwendig wäre, da ohne Verbindung zum Server der NC ein "totes Gerät" ist. Während in den USA dies wenigstens noch eine in Betracht zu ziehende

Möglichkeit darstellt (in den USA sind Ortsgespräche meist kostenlos), würde in Europa und anderen Länder wohl niemand ernsthaft an diesen Einsatzzweck denken.

Der NC ist also im Grunde nichts anderes als ein Terminal, das sich besonders im Inter- oder Intranet bewährt. Die Funktionalität entspricht der eines herkömmlichen PCs im Inter-/Intranet-Bereich. Um die Idee des NCs durchzusetzen, bildeten Apple, IBM, Netscape, Oracle und Sun ein Konsortium, welches das "NC Referenz Profile", eine Beschreibung der notwendigen Hard- und Software, verabschiedete.

3.2 Network Computer in der Realität

Mittlerweile wird fast von Jedem ein neuer NC mit Client/Server-Struktur angekündigt. Einige Geräte haben bereits die Marktreife erreicht, wenige sind sogar schon erwerblich. Allerdings basieren nur wenige der angekündigten Geräte auf dem NC Referenz Plattform Profile. Nicht alles, was heute als NC angepriesen wird, ist das, was es zu sein vorgibt. Nachdem ein Wort wie Network Computer allein nicht sehr viel aussagt, verwenden viele Hersteller dieses Modewort, um alte Geräte unter dem Deckmäntelchen eines innovativen Gerätes auf den Markt zu werfen. Für den verunsicherten Anwender wird es immer schwieriger zu erkennen, ob es sich bei einem Angebot auch tatsächlich um das handelt, was einen NC ausmacht. Es läßt sich zwischen drei Arten von Angeboten unterscheiden:

3.2.1 Die konventionelle Network Computer

Echte NCs, also solche, die sich streng am NC Referenz Profile orientieren, findet man momentan nur bei den großen renommierten Hersteller der Computerbranche (Sun, IBM, Apple,). SUN zeigt, wie ein richtiger NC auszusehen hat: SPARC-II mit 110 MHz, auf Java optimiert (das JavaOS braucht nur 3,5MB Arbeitsspeicher). Die meisten NCs können allerdings wesentlich mehr als verlangt. So möchte IBM seine Network Station (NS) an Arbeitsplätzen etablieren, an denen bisher IBM-Terminals standen. Die NS vereinigt also die Funktionalität eines NCs, eines X-Terminals und mehrerer herkömmlicher textbasierender Terminals. Als Server empfiehlt IBM einen Mainframe (z.B. RS6000) als beste Lösung. Software-Lösungen für "die Echten" (also Java-Anwendungen) wurden schon von Oracle und Corel vorgestellt. Von Corel kann man sogar schon eine Beta-Version von "Corel Office für Java" herunterladen.

3.2.2 Die Windows-Network Computer

Von der Firma Citrix kommt eine andere Idee für Thin Clients. Winframe ist eine Erweiterung für den Microsoft NT Server (Version 3.51 und 4.0), die diesem die fehlende Multiuserfähigkeiten beibringt. Die Software ist unter dem Namen MS NT Terminal-Server 4.0 zu lizensieren. Die Software stellt die Ausgabe der zentral auf dem NT-System laufenden Anwendungen auf den "Windows-NCs" dar. Der NC ist in diesem Fall nichts anderes als ein „dummes“ Terminal. Im Vergleich zu der bisherigen Methode (NT-Server

und daran hängender Windows PCs) kann dieses System in einigen Punkten dennoch überzeugen. Gerade in Bereichen, in denen Sicherheit in Bezug auf die Virenproblematik und auch Ausfallsicherheit auf der einen Seite und einfache Installation und Wartbarkeit auf der anderen Seite wichtig sind, besonders hohe Leistung aber nicht gefragt ist, kann das Winframe Client/Server-System eine elegante Lösung sein. Weiterhin ist (fast) 100% Kompatibilität zu den Standardsoftwareprodukten gewährleistet. Verfügbar sind Geräte in diesem Bereich bereits z.B. von Boundless (Network Computer TC/XL/XLC), dem Terminalhersteller Wyse (WinTerm 2500T/2700T) und Esesix (NT/Net).

3.2.3 Die Marketing Network Computer

Die hier "Marketing NC" genannten Geräte haben mit einem NC im eigentlichen Sinne nichts gemeinsam. Vielmehr sind es gewöhnliche PCs (also Intel Prozessor und Microsoft Windows als Betriebsystem), die aus Marketinggründen den Namen NC tragen. Geräte dieser Gattung kommen vor allem aus den großen Diskontläden. Allerdings verzichten leider auch einige der namhafteren Firmen des Computerbereichs nicht auf diese unseriöse Art der Werbung. Siemens bietet z.B. lediglich eine Sparvariante ihrer SNI-PC Reihe ohne Festplatte, jedoch mit Windows-Oberfläche und Boot-ROM als NC an. Zur Sicherheit kann man ihn noch zu einem "normalen" PC aufrüsten. Mit einer 180 Grad-Wendung überraschte auch die bisherige Anti-NC-Front "Wintel" (Windows und Intel). Auch sie wollen jetzt einen NC verkaufen: 100MHz Pentium, 16MB, Windows95 bzw. NT

3.3 IBM Network Computer

In dieser Arbeit werden zwei Netzwerk Computer verwendet: ”Network Computer Reference Platform“ und „Network Station”. Die NS ist ein auf dieser ersten Referenz-Plattform weiterentwickelter Netzwerk Computer. Diese NS wurde um einiges wie PCMCIA Schnittstelle (Personal Computer Memory Card International Association) erweitert. Obwohl beide Plattformen, bis auf einige Kleinigkeiten, die gleiche Struktur besitzen, werden sie in zwei unabhängigen Kapiteln behandelt. Diese Arbeit beschäftigt sich überwiegend mit NC.

4 Network Computer Referenz Plattform

Dieses Kapitel umfaßt Vorstellung, Software, Inbetriebnahme und aufgetretene Probleme des NCs.

4.1 Network Computer Referenz Plattform

Nach dem ersten Schritt in Richtung Definition eines NCs (August 1996, durch die Firmen Apple, IBM, Oracle, Netscape und Sun) wurde ein NC mit PowerPC-Architektur von IBM vorgestellt, der fast alle grundlegenden Spezifikationen unterstützt. Die Referenz Plattform ist auf Internet und Java aufgebaut. Auf der Systemnahen Ebene verfügt der NC über die Standards TCP, HTTP, HTML, Java Anwendungsumgebung und einige grundlegende IP-Protokolle wie UDP und SNMP. Darüber hinaus ist die Unterstützung von Internet-Mail (einschließlich der Protokolle POP und IMAP) und einiger gängiger Graphik- (GIF, JPEG) und Audioformate (Windows WAV, AU von Sun) gefordert. Der NC muß auch eine "Zeigevorrichtung" haben, mit der man z.B. eine graphische Oberfläche bedienen kann. Auch die Möglichkeit der Texteingabe, eine Mindestgröße des Bildschirms von 640 zu 480 (VGA) und Audio-Wiedergabe sind festgelegt. Ein NC kann um zusätzliche Protokolle, etwa FTP, Telnet oder NFS (Network File System), und Sicherheitsmerkmale wie Netscapes SSL für sichere Verbindungen, den ISO-Standard 7816 (Smart Card) oder die Master Card/Visa-Spezifikation für Zahlungsvorgänge erweitert werden. Ebenfalls nur als Option ist die Fähigkeit zum Drucken vorgesehen.

Abbildung 4.1: Thin Client [12]

Diese Grundspezifikation scheint ziemlich selbstverständlich, und doch sticht ein Punkt ins Auge: Die Richtlinie, daß keine permanente lokale Speicherung benötigt wird. Dadurch ergeben sich für die Administration neue Möglichkeiten, die es erlauben, sowohl das Betriebssystem als auch die Applikationen laufend auf dem neuesten Stand zu halten, ohne daß die Benutzer damit zu tun haben.

Der als Zielsystem verwendete NC enthält die ganze Hardware, Monitor und Netzteil, d.h. der NC und andere Komponenten sind in einer Kompaktform (Industrie-PC)

zusammengebaut [12]. Als Monitor wird eine LCD-Anzeige verwendet. Auf der Bedienungstafel sind Tastatur und Trackmaus eingebracht. Die Abbildung 4.1 zeigt die gängige Form des Industrie-PCs. Für bequemes Arbeit wird jedoch eine externe Tastatur über ein Spiralkabel mit 5poligem Dioden-Stecker (DIN) angeschlossen.

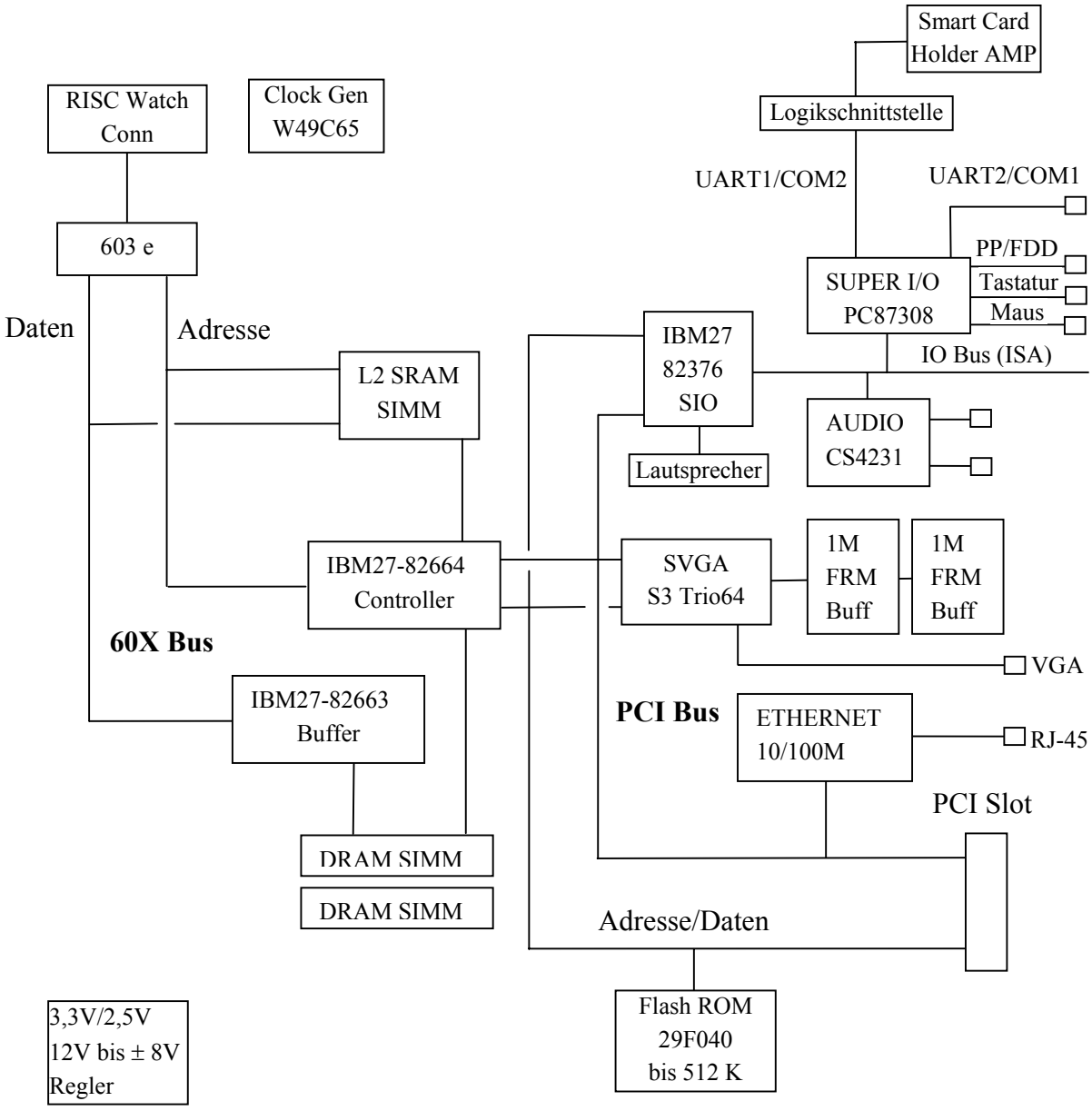


Abbildung 4.2: Block Diagramm der NCRP [7]

Die Hauptmerkmale der Referenz Hauptplatine (PowerPC Board) sind folgendermaßen zu beschreiben:

- CPU PPC 603eV mit 200 MHz Taktfrequenz
- Arbeitsspeicher 64 MB DRAM, 512 KB L2 Cache, 5MB Flash ROM
- Ethernet 10/100 M, 1 parallele und 1 serielle Schnittstelle, eine PCI-Bus Schnittstelle
- Smart Card Lesegrtät mit T1 Hardware Unterstützung

Das Block Diagramm ist in Abbildung 4.2 zu sehen.

4.2 Boot-Methoden

Da ein NC über keine Platte verfügt, von der gebootet werden kann, muß er die Informationen auf eine in der folgenden Weisen abrufen:

- DHCP (Dynamic Host Configuration Protocol) über Ethernet Anschluß (Ethernet DHCP)
- Floppy; da jedoch der erforderliche Speicherplatz für das vorgesehene Betriebssystem (JavaOS) größer als 6MB ist, wird die Kapazität der Floppy nicht ausreichen
- IDE Plattenlaufwerk

4.3 Erforderliche Software auf dem Server

Als Server kommen alle herkömmlichen Server in Frage, wobei allerdings die folgenden Dienste darauf installiert und gestartet sein sollen: DHCP-Server, TFTP-Server und Internet Information Server. Hier wurde ein NT-Terminal Server 4.0 eingesetzt, um über JICA-Java-Client auf MS-Anwendungen auf dem NT-Server zuzugreifen. Immerhin fehlt auf dem NT-Terminal Server das Winframe Paket.

Mit DHCP (ein TCP/IP-Protokoll) kann man eine IP-Adresse und Konfigurationsdaten automatisch zuordnen, ohne die MAC-Adressen der Rechnern notieren und verfolgen zu müssen. Mit DHCP kann jedem Host einem zuvor festgelegten Bereich von IP-Adressen entweder eine permanente oder temporäre IP-Adresse zugeordnet werden. In dieser Arbeit wird ein von der Firma **hughsystem** [18] speziell für IBM Net Center unter Java entwickelter DHCP-Server eingesetzt.
TFTP-Server überträgt Dateien zwischen Server und Clients. Ein TFTP-Server wird beim Starten eines Computers ohne Festplatte verwendet. TFTP wurde als kleines und einfaches Protokoll entworfen, damit für den Client Nur-Lese-Rechte eingerichtet werden können. TFTP verwendet die UDP-Transportschicht (und nicht die TCP-Transportschicht). Hier wurde zunächst das von **hughsystem** empfohlene **tftppro** [28] eingesetzt. Danach sind auch andere TFTP-Server auf dem NT-Server installiert und verwendet worden. In keinem Fall konnte eine schnelle Datenübertragung erreichen werden. Zur Zeit braucht der TFTP-Server ca. 50 Sekunden, um Boot-Daten (JavaOS) zum NC zu transportieren. Die beiden

Server sind nicht als Dienst sondern nur als normal ausführbare Programme installiert. Deshalb muß man sie manuell ausführen und beenden.

4.3.1 Konfiguration der DHCP- und TFTP-Server

Bevor man die Dienste startet, muß man sich der Konfiguration der Server widmen. Zuerst wird die Konfiguration von DHCP kurz erläutert. Das Config-Verzeichnis (**C:\Program Files\dhcp\config**) enthält drei Properties (Eigenschaften), die bislang nur für diesen NC bearbeitet worden sind:

4.3.1.1 dhcpdefault.properties

Dieses File beinhaltet alle Informationen, die für DHCP-Server nötig sind. Ein Teil davon ist die IP-Adresse, die zwischen *.start und *.end abgegrenzt ist. Alle Properties können durch ein macro überschrieben werden. Abbildung 4.3 zeigt dieses Konfigurations-Skript:

```
############## The following properties cannot be overridden by a macro ###############

dhcp.iprange.start = 111.110.1.6
dhcp.iprange.end = 111.110.1.15

################ The following properties can be overridden by a macro ################

# General properties
dhcp.leasetime = -1

# IP parameters
dhcp.subnetmask = 255.255.255.0
dhcp.router =

# Boot properties
dhcp.bootserver = 111.110.1.5
dhcp.bootfile=javaOS

# Network services
dhcp.timeserver = 111.110.1.5
dhcp.dnsserver = 111.110.1.5
dhcp.dnsdomain = 111.110.1.5

# Vendor specific options
#Diese Optionen sind in File netcenter.properties zu finden.
#########################################################################
########
```

Abbildung 4.3: Konfigurations-Skript; dhcpdefaults.properties

Im folgenden werden die Optionen kurz beschrieben.

leasetime:	Die Zeit in Sekunden, die der Client die IP-Adresse verwenden kann. Wenn –1 verwendet wird, ist die Pachtzeit endlos.
subnetmask:	Netmask-Adresse
router:	Router-Adresse (optional)
bootserver:	IP-Adresse des Bootservers
bootfile:	Das File, das über TFTP in NC transportiert wird
timeserver:	Die IP-Adresse eines Servers, der das NTP (Network Time Protocol) unterstützt. (RFC 2030)
dnsserver:	IP-Adresse des DNS (Domain Name System) Servers
dnsdomain:	DNS Domain-Name des Servers

4.3.1.2 dhcpserver.properties

In diesem Skript kann man einen Logfilepfad angeben und eine Loginanforderung ein- und ausschalten. Hier ist der Makro dem NC als *netcenter* mit seiner festen Hardware-Ethernet Adresse (MAC) zugewiesen. Damit können die Properties in *dhcpdefaults.properties* überschrieben werden. Die MAC-Adresse in Abbildung 4.4 gehört zu der NC Referenz Plattform. Mit der Zeile ⇦(*.*netcenter=MAC-Adresse*) wird der Makro für einen NC definiert.

```
#DHCP Server Properties file
# General properties
#dhcp.log = on
#dhcp.logfile = null
#Ethernet addresses and macro names
dhcp.macro.netcenter = 0004aceb020b ⇦
#dhcp.macro.mbx = 08003e262944
#dhcp.macro.ads = 0000a72301cc
```

Abbildung 4.4: Auszug aus dem dhcpserver.propertiesAbbildung

4.3.1.3 netcenter.properties (macro)

JavaOS verwendet dieses File für seine Umgebung wie z.B. Druckeraktivierung, Zeitzone und Starten von Anwendungen. Die System- und JavaOS-Properties in 4.4.2 werden später erläutert [9].

```
#dhcp.bootserver = 111.110.1.5
#dhcp.bootfile = javaOS
dhcp.vendor.JavaOS.Generic.101 = "-djavaos.apps=http://111.110.1.5/JavaOSApps.html"
dhcp.vendor.JavaOS.Generic.102 ="-djavaos.login=false -djavaos.console=false"
dhcp.vendor.JavaOS.Generic.103 ="-djavaos.netShell=true -djavaos.printservice.local.params.serial-COM1=9600:8:1:none"
dhcp.vendor.JavaOS.Generic.104 ="-djavaos.printservice.local.params.parallel-LPT1=disabled"
```

Abbildung 4.5: Inhalt der Datei netcenter.properties

4.3.1.4 TFTP-Konfiguration

Bevor der TFTP-Server verwendet werden kann, müssen auf dem Server die Verzeichnisse, auf die ein Zugriff erlaubt ist, gekennzeichnet werden. Für den TFTP-Server sind zusätzliche Parameter vorhanden, doch handelt es sich dabei um Parameter mit Standardwerten, die häufig nicht geändert werden müssen. Der Server ist im Verzeichnis **C:\Program Files\Walusoft** installiert. Ebenso ist auf dem Desktop eine Verknüpfung auf TFTPServer32.exe Datei verfügbar. Zur Parametereinstellung startet man den TFTP-Server und klickt im Dialogfeld nacheinander auf System→Setup. Im Register **Outbound** muß der Pfad der zu transportierenden Datei angegeben werden, in diesem Fall: **C:\Program Files\dhcpserver\tftpboot**. Die anderen Register bleiben unberührt. Weitere Informationen können Online aufgerufen werden.

4.4 Booten des NCs

Nachdem die Konfiguration der Server beendet ist, soll das Bootprogramm (Firmware) aufgerufen werden. Wenn die Einstellungen nicht richtig sind, wird der Aufruf des BIOS automatisch verlangt, indem man eine willkürliche Taste drückt. In dem Main Menü ist die System Konfiguration Menü sehr wichtig. Damit werden die hauptsächlichen Einstellungen durchgeführt. Hier werden nur die für diese Arbeit nötigen Attribute erläutert, der Rest ist der Literatur [7] zu entnehmen. Nach der Aktivierung von ”System Configuration Menu→System Information” werden die Hauptmerkmale des Gerätes angezeigt. Sie sind nochmals in Abbildung 4.6 zusammengeführt.

System Processor	PowerPC 603eV
Installed Memory	64 MB
Second-Level Cache	512 KB, Burst, Writethrogh, HW, Invalidate
Upgrade Processor	Not Installed
Ethernet MAC Address	00:04:AC:EB:02:0B
Firmware Revision	1.0
Firmware Build Number	97071107

Abbildung 4.6: System Information des NCs

Mit der ”System Configuration Menu→Configure I/O Devices” kann man die üblichen Informationen, die während des Bootvorgangs auf der Standardausgabe ermittelt werden, auf einen externen Rechner über serielle Schnittstelle weiterleiten. Auf dem Zielrechner soll ein einfaches Programm wie ”SerialDemo” von Sun [20] gestartet sein. Sehr wichtig ist hier die Ziele ”Set Boot Devices”. Obwohl die ”Firmware” auf dem NC und in der Einleitung [7] die gleiche Version aufweisen, unterstützt die programmierte ”Firmware” einige oder mehrere Protokolle. Wie aus Tabelle 4.1 ersichtlich ist, sind die Hauptunterschiede bei der Netzkarte festzustellen. Hier muß **ETHERNET DHCP** ausgewählt werden. Als letztes Attribut wird die Umstellung von ”**Debug Mode**” auf **”[Debug On]**” empfohlen, damit man die Fehler besser lokalisieren kann.

Firmware auf dem NC		**Firmware in [7]**	
Firmware Revision	1.0	Firmware Revision	1.0
Firmware Build Number	97071107	Firmware Build Number	97020515
1. ETH_NET BooTP		1. ETHERNET AUTO	
2. TokenRG BooTP		2. TOKENRG AUTO	
3. ETHERNET DHCP		3. Floppy 1	
4. TokenRG DHCP		4. Floppy 2	
5. Floppy 1		5. IDE Drive 0	
6. Floppy 2		6. IDE Drive 1	
7. IDE Drive 0		7. None	
8. IDE Drive 1			
9. None			

Tabelle 4.1: Vergleich der Firmware des NCs

Hinweis:
Durch einem technischen Fehler wird die Knopfbatterie in kurzer Zeit entladen, so daß deshalb vor jedem Start des NCs die Attribute neu eingestellt werden sollten.

4.4.1 Inbetriebnahme des NCs

Bevor man den NC einschaltet oder vor dem Reboot muß man auf dem Server den DHCP - und den TFTPpro-Server starten. Nach dem üblichen Verbindungsaufbau zwischen Server und NC wird die JavaOS-Binärdatei über TFTPpro-Server zu dem NC transportiert. Nachdem das Betriebssystem entpackt ist, wird eine Html-Seite ausgeführt (siehe 4.4.3). Während des Entpackens und der Selbstkonfiguration des JavaOS können die Nachrichten mittels ”Console” auf der Standardausgabe verfolgt werden. Das Nachrichtausgabe-programmm ”Console” wird durch Drücken der Druckertaste auf der Bedienungsfläche aufgerufen. Das ist ein ”JavaOS.Property”, das man umstellen kann [9].

```
<HTML>
<HEAD>
  <META  HTTP-EQUIV="Content-Type"  CONTENT="Content-Type"  CONTENT="text/html;
charset=iso-8859-1">
  <META NAME="GENERATOR" CONTENT="Mozilla/4.03 [en] (WinNT; I) [Netscape]">
<TITLE>JavaOS Applications</TITLE>
</HEAD>
<BODY>
<BR> 
<application code="BlackBox" archive="http://111.110.1.5/nc_prg/BlackBox.zip" >
<param name ="name" value="BlackBox">
</application
<BR> 
<application                                                    code="ParallelBlackBox"
archive="http://111.110.1.5/nc_prg/ParallelBlackBox.zip" >
<param name ="name" value="ParallelBlackBox">
</application
<BR> 
<application code="com.citrix.JICA" archive="http://111.110.1.5/JICA.zip" >
<param name ="name" value="ADS-tec - Java Client on JavaOS ">
<param name ="args" value="-address:111.110.1.6 -width:640 -height:480 -borderwidth:0">
</application>
<BR> 
<application code="QuadraticGraph" archive="http://111.110.1.5/nc_prg/Quadratic.zip" >
<param name ="name" value="QuadraticGraph">
<param name ="args" value="1 2 3">
</application>
<BR> 
<application code="Syste" archive="http://111.110.1.5/nc_prg/systempropert.zip" >
<param name ="name" value="Systemproperties">
</application>
<BR> 
<application code="SerialDemo" archive="http://111.110.1.5/nc_prg/SerialDemo3.zip" >
<param name ="name" value="SerialDemo">
</application>
<BR> 
<application code="SimpleRead" archive="http://111.110.1.5/nc_prg/SimpleRead.zip" >
<param name ="name" value="SimpleRead">
</application>
<BR> 
<application code="SimpleWrite" archive="http://111.110.1.5/nc_prg/SimpleWrite.zip" >
<param name ="name" value="SimpleWrite">
</application>
<BR> 
<application code="sunw.hotjava.Main" archive="http://111.110.1.5/hotjava.zip">
<param name ="name" value="HOT JAVA BROWSER">
<param name ="homeprop" value="hotjava.home">
<param name ="args" value="-width:1024 -height:768">
<param name ="args" value="http://111.110.1.5/Citrix/JClient.html">
</application>
</BODY>
</HTML>
```

Abbildung 4.7: Sourcecode der JavaOSApps.html-Seite des Applaunchers

4.4.2 JavaOS Properties

Es gibt zwei Arten von Properties: System- und JavaOS-Properties. Systemproperties sind auf allen Java-Plattformen durch die JDK System.getProperties() Methode erhältlich.

JavaOS-Properties sind nur als Code erhältlich und werden von JavaOS selbst für seinen Aufbau verwendet. JavaOS Codes können durch die *getProperties()* Methode der *sun.javaos.Env* Klasse auf Properties zugreifen. Properties können auf sechs verschiedene Arten definiert werden. In dieser Arbeit wird die JavaOS shell und DHCP Befehlszeile benutzt, um JavaOS richtig zu konfigurieren. Zum Beispiel wird durch *djavaos.console=true* eine interaktive Befehlszeile auf einem anderen Rechner realisiert (das ist möglich, wenn COM-Anschlüsse der beiden Rechner mittels eines Nullmodemkabels miteinander verbunden sind). Mit der Eingabe von **help** werden alle Shell-Befehle (**commands**) aufgelistet.

Wenn man auf eine serielle Schnittstelle zugreifen will, muß diese Property auf false gesetzt werden. Sonst können Anwendungen wie SerialDemo (ein einfaches Terminalprogramm) den Port nicht öffnen. Die Anwendungsausführung wird mit der folgenden Fehlermeldung unterbrochen: *"Auf COMx kann nicht zugegriffen werden. Der Port wird von anderer Anwendung benutzt."*

4.4.3 Laden der Applikationen in JavaOS

Es gibt drei Wege, eine Applikation in JavaOS zu laden:

- Die Anwendung kann in JavaOS aufgebaut und zusammen mit ihm im ROM gespeichert werden.
- Die Anwendung kann sich auf einem Server befinden und automatisch nach dem Bootvorgang in einer JavaOS-Sitzung gestartet werden.
- Die Anwendung kann vom Benutzer auf einem graphischen Dienstprogramm (Namens **"AppLauncher"**) aufgerufen und vom Server zum NC geschickt werden.

Hier wird die letzte Methode verwendet. Dazu werden Applikationen von einem graphischen Dienstprogramm, das **"AppLauncher"** genannt wird, ausgewählt, und als Zip-Datei vom Server zum Client transportiert (über TFTP) und gestartet. Die JavaOSApps.html Datei (**C:\InetPub\wwwroot**), die als **"AppLauncher"** durch JavaOS gestartet wird, ist in Abbildung 4.7 zu sehen. Die folgenden Begriffe werden in "JavaOSApps" verwendet.

code	die Hauptklasse der Anwendung
archive	Pfad der Zip-Datei der Anwendung
name	der Name der Anwendung, der im AppLauncher-Fenster gezeigt wird

param homeprop	property, den die Anwendung verwendet, um ihr Fileroot anzusprechen
param args	Liste der durch Leerzeichen getrennten Argumente für *main()*
param icon	URL von einem Bild, das als Icon für die Anwendungspräsentation gilt

4.5 JavaOS und Java Communications API

Die Untersuchung der Software-Einbindung in JavaOS während der Grundlagenanalyse hatte zum Ergebnis, daß die Integration von Hardwarezugriffsanwendungen wie z.B. serielle Schnittstelle nur durch eine zusätzliche Comm API möglich sind. Der Hauptgrund dafür liegt in der JavaOS-Architektur und Java.

4.6 Hardwarezugriff unter Java und JavaOS auf dem NC

Es wurde am Anfang dieser Arbeit angenommen, daß das Paket javax.comm der Comm-API (laut SUN-Veröffentlichung) im JavaOS-Betriebssystem enthalten ist. Danach sind einige Versuche von selbst entwickelten Anwendungen durchgeführt worden. Die ersten Programme konnten nicht gestartet werden, da das Argument für diese Programme (in JavaOSApps.html) falsch war. Als erstes Beispiel wurde das Programm SystemProperties und danach QuadraticGraph (eine quadratische Funktion mit Eingabeparameter) ausgeführt. Danach wurden verschiedene Anwendungen zur Überprüfung der Funktionalität der seriellen Schnittstellen durchgeführt. Diese Anwendungen sind als Zip-Datei im WWW-Rootverzeichnis abgelegt. Die JavaOSApps.html Seite wurde entsprechend bearbeitet. Zur Zeit ist in dem AppLauncher-Fenster die folgende Liste der Anwendungen und Browser zu sehen:

BlackBox (nicht funktionsfähig), ParallelBlackBox (nicht funktionsfähig), SerialDemo, SimpleRead, SimpleWrite, QuadraticGraph, Systemproperties, ADS-tec-Java Client on JavaOS (auf dem NT-Server fehlt Winframe) und HOT JAVA Browser
Da die Anwendungen für die serielle Schnittstelle nach mehreren Versuchen nicht erfolgreich durchgeführt werden konnten, wurde anhand der Fehlermeldungen nach den fehlenden Methoden und Klassen gesucht. Als Beispiel werden die Fehlermeldungen nach dem Aufruf der Anwendungen SimpleRead und SerialDemo unten aufgezeigt:

launching application: SimpleRead
PPc Pcnet.Receive: crc error

Launching SerialDemo died with exception!
Java.lang.NoclassDefFound Error: Serial Parameters
At SerialDemo.<init> (SerialDemo.java)
At SerialDemo.main (SerialDemo.java)
At Sun.javaos.application.Main (SerialDemo.java)
At java.lang.Thread.run()

Nachdem der SerialDemo-Quellcode unter JDK 1.1.4 compiliert wurde (zuerst war der Quellcode unter JDK1.1 und 1.1.5 compiliert), ist SerialDemo teilweise gestartet, d.h. nach dem Start konnte man den Port nicht öffnen. Dieses Problem ist einfach zu lösen, indem man JavaOS-Property für ”console” (siehe Abbildung 5) in ”false” einsetzt. Immerhin war kein Empfang von Zeichen möglich.

Unter Berücksichtigung der Fehlerausgabe ist der Aufruf der Methode ”getEventType” auskommentiert. Mit dieser Änderung ist es gelungen, die ankommenden Zeichen aufzufangen. In Abbildung 4.8 sind die Zeilen ersichtlich.

```
...
} catch (InterruptedException e) {}
    }
    public void serialEvent(SerialPortEvent event) {

//        switch(event.getEventType()) {
              //      case SerialPortEvent.BI:
             //        case SerialPortEvent.OE:
             //        case SerialPortEvent.FE:
             //        case SerialPortEvent.PE:
             //        case SerialPortEvent.CD:
             //        case SerialPortEvent.CTS:
             //        case SerialPortEvent.DSR:
            //         case SerialPortEvent.RI:
//        case SerialPortEvent.OUTBUF_EMPTY:
//            break;
//        case SerialPortEvent.DATA_AVAIL:

            byte[] readBuffer = new byte[20];
            try {
                while (inputStream.available() > 0) {    ...
```

Abbildung 4.8: der auskommentierte Quellcode von SimpleRead.java

Entsprechend ist auch dieser Methodenaufruf in der SerialConnection.java (eine Klasse zur Behandlung der Details eines Serialports) auskommentiert.

Anhand der Dokumentation von JavaOS in der Entwicklungsphase stellt man fest, daß viele Merkmale des Betriebssystems umbenannt oder abgeschafft wurden [9]. Das *javax.comm* Paket ist sicher in JavaOS vorhanden, aber ein Teil des Pakets wird nicht unterstützt. Da die Unterlagen von der aktuellsten Version (JavaOS 1.1.1) fehlen, ist keine

zuverlässige Aussage möglich. Der Entwickler kann diese Comm API verwenden, um Anwendungen in Bezug auf die serielle oder parallele Schnittstelle zu schreiben.

5 Network Station, OS-9000 und JVM

Der Netzcomputer IBM Network Station bietet eine attraktive Alternative zu den herkömmlichen Methoden des Network Computing. Einzelne Datenstationen ohne Plattenspeicher werden mit einem Server (oder einer Reihe von Servern) verbunden und zentral mit dem Programm IBM Network Station Manager (NSM) verwaltet.

Der Hauptunterschied zu einem PC besteht darin, daß sich die NS-Dateien auf einem Netzserver und nicht auf einer Festplatte der einzelnen Benutzermaschinen befinden. Die NS arbeitet mit einer graphischen Benutzerschnittstelle (GUI), über die der Benutzer auf zahlreichen Ressourcen zugreifen kann. Mit NS ist der Zugriff auf die folgenden Ressourcen möglich:

- 5250-Emulator
- 3270-Emulator
- Web-Browser
- Java-Applets oder –Anwendungen
- Windows NT-Anwendungen
- Lokale und ferne Drucker
- X-Anwendungen auf fernen Systemen

Die NS kommuniziert mit Hilfe von TCP/IP über ein Ethernet mit dem Server. Auf jeder NS wird das gemeinsame Client-Programm und auf dem Server werden das Programm NSM und verschiedene andere Anwendungsprogramme ausgeführt.

Der Netzwerk Computer Serie 1000 (8362-A53) ist als Java-System positioniert. In ihm entfaltet ein PowerPC 603e mit 200 MHz seine Leistung. Zunächst wird die Hardware kurz vorgestellt.

Dieses Modell verfügt über zwei SIMM-Sockel, die mit 64 MByte ausgestattet sind, sowie über eine S3-TRIO-Graphikeinheit, die mit 2 MByte VRAM versehen ist. Damit sind Auflösungen von 640*480 bis 1600*1200 SXGA bei 256 möglich. Die Hardware unterstützt 10/100 MBps (Ethernet) über einen RJ45 (10BaseT) Anschluß.

Außerdem befinden sich auf der Rückseite Anschlüsse für je eine serielle, parallele und PCMCIA Schnittstelle. Auf der Vorderseite sind Anschlüsse für Kopfhörer, Lautsprecher (16-Bit-Audio-System), Smart Card, Tastatur und Maus verfügbar. Die NS verfügt über eine Zwei-Tasten-Maus und eine 102-Tasten-Tastatur. Der Leser findet in [8] weitere Informationen.

Beim Einschalten der NS geschieht folgendes:

- Die NS führt zunächst eine Hardwarediagnose durch.
- Die NS baut eine Übertragungssitzung zum zugeordneten Server auf.
- Die NS lädt den Betriebssystemcode vom Server in den Arbeitsspeicher (RAM).

Nachdem der Betriebssystemcode in die NS geladen wurde, erscheint eine Anmeldeanzeige. In dieser Anzeige werden Informationen zum System, zur Benutzer-Identität (ID) und zum Kennwort angefordert. Die NS prüft die Benutzerinformationen und lädt die persönliche Systemkonfiguration sowie Vorgaben herunter wie beispielsweise die Anwendungsdarstellung oder die Tastaturbelegung. Eindeutige Vorgaben können auf jede NS, die auf dem lokalen Netz (LAN) verwendet wird, heruntergeladen werden. Die NS kommuniziert dann mit anderen berechtigten Servern.

5.1.1 Inbetriebnahme

Zunächst sollen die Seriennummer, der Maschinentyp, die Modellnummer und der installierte Speicher in eine Tabelle eingetragen werden. Damit wird dem Benutzer eine einfache Referenz zur Konfiguration und Wartung ermöglicht. Die in dieser Arbeit verwendete NS hat die in Tabelle 5.1 eingetragenen Daten.

Kategorie	Eigene Angaben
Seriennummer	23MB996
Maschinentyp und Modellummer	8362-A53
MAC-Adresse	00:00:A7:23:01:CC
Installierter Speicher	64 MByte (DRAM)

Tabelle 5.1: Einträge der NS

Für die Hardwareinstallation sind die folgenden Teile notwendig:

Logikeinheit (NS), Stromversorgungsmodul, Tastatur (PS2), Maus (PS2), Bildschirm (mind. 640*480) und 10BaseT-Verbindungskabel mit RJ-45-Stecker.
Beim Anschließen obiger Teile an die NS kann die Abbildung 5.1 zu Hilfe genommen werden.

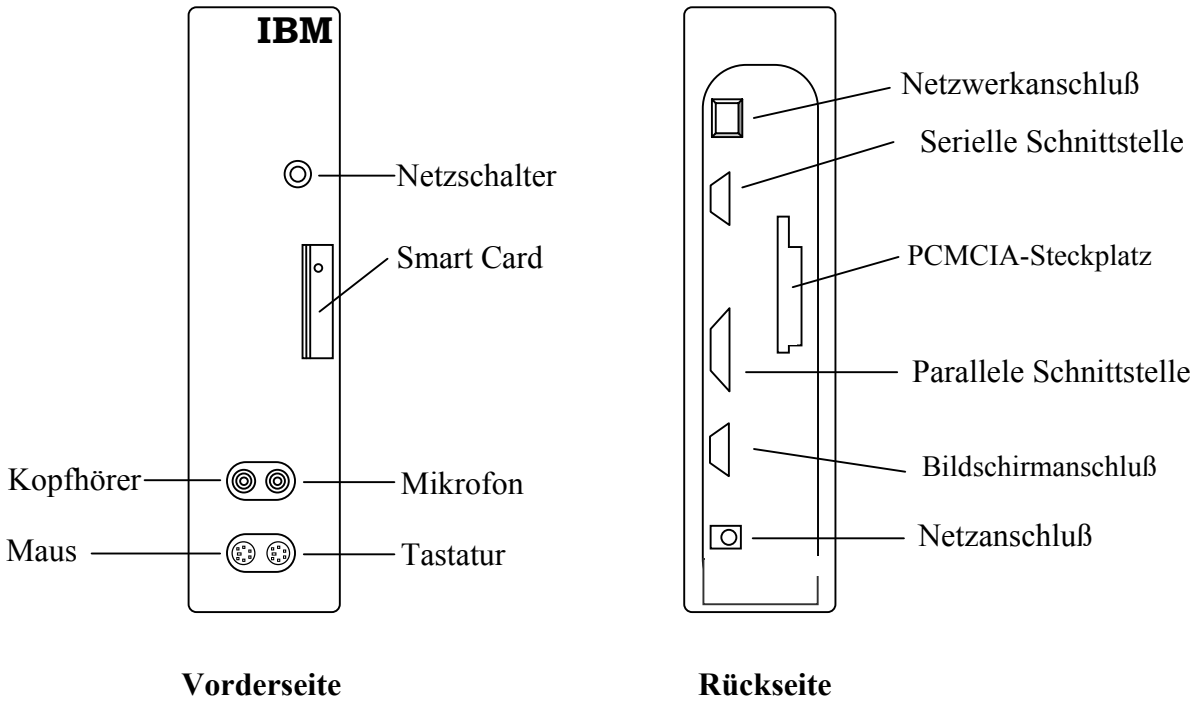


Abbildung 5.1: Installation der Teile auf der Vorder- und Rückseite

5.1.2 Booten der Maschine

Nach dem Einschalten der NS werden die unten erläuterten Schritte durchgeführt.

In dieser Arbeit ist letztendlich nur ein NT-Server eingesetzt worden. Die Reihenfolge ändert sich dadurch nicht. Die Dauer des GUI-Aufbaus hängt vom Server-Typ und von der Anzahl der verwendeten Server ab. Wie aus Abbildung 5.2 ersichtlich ist, werden folgende Dienste ausgeführt.

1. Ein im nicht flüchtigen Arbeitsspeicher (NVRAM, Non-Volatile Random Access Memory) befindliches Boot-Überwachungsprogramm (Boot-Monitor) wird gestartet. Daraufhin führt die NS automatisch eine Reihe von Selbsttests beim Einschalten (POST, Power-On Self Test) aus.

2. Die NS stellt eine Verbindung zu einem BOOTP (BOOTstrap Protocol)- oder DHCP (Dynamic Host Configuration Protocol)-Boot Server her. Die NS übergibt ihre MAC-Adresse (Media Access Control) an den Server und erhält daraufhin eine IP-Adresse vom Server. Der Boot-Server stellt ferner die Adresse oder den Pfad des Basiscode-Servers zur Verfügung. Als Alternative hierzu kann die NS diese Informationen auch aus den in ihrem NVRAM gespeicherten Daten abrufen.

3. Die NS lädt den Basiscode per TFTP (Trivial File Transfer Protocol) oder NFS vom Basiscode-Server herunter.

4. Die NS lädt die Konfigurationsdaten für die Datenstation vom Server für die Datenstationskonfiguration herunter.

5. An der NS erscheint eine Anmeldeanzeige. Nach Eingabe einer Benutzer-ID und eines Kennworts durch den Benutzer, überprüft der Authentifizierungs-Server die Identität des Benutzers.

6. Der Konfigurations-Server des Benutzers lädt die persönlichen Umgebungsvorgaben des Benutzers herunter und initialisiert diese.

7. Die NS zeigt die persönliche Arbeitsoberfläche des Benutzers an. Der Benutzer kann dann bei den jeweiligen Anwendungen auf die verschiedenen Server zugreifen.

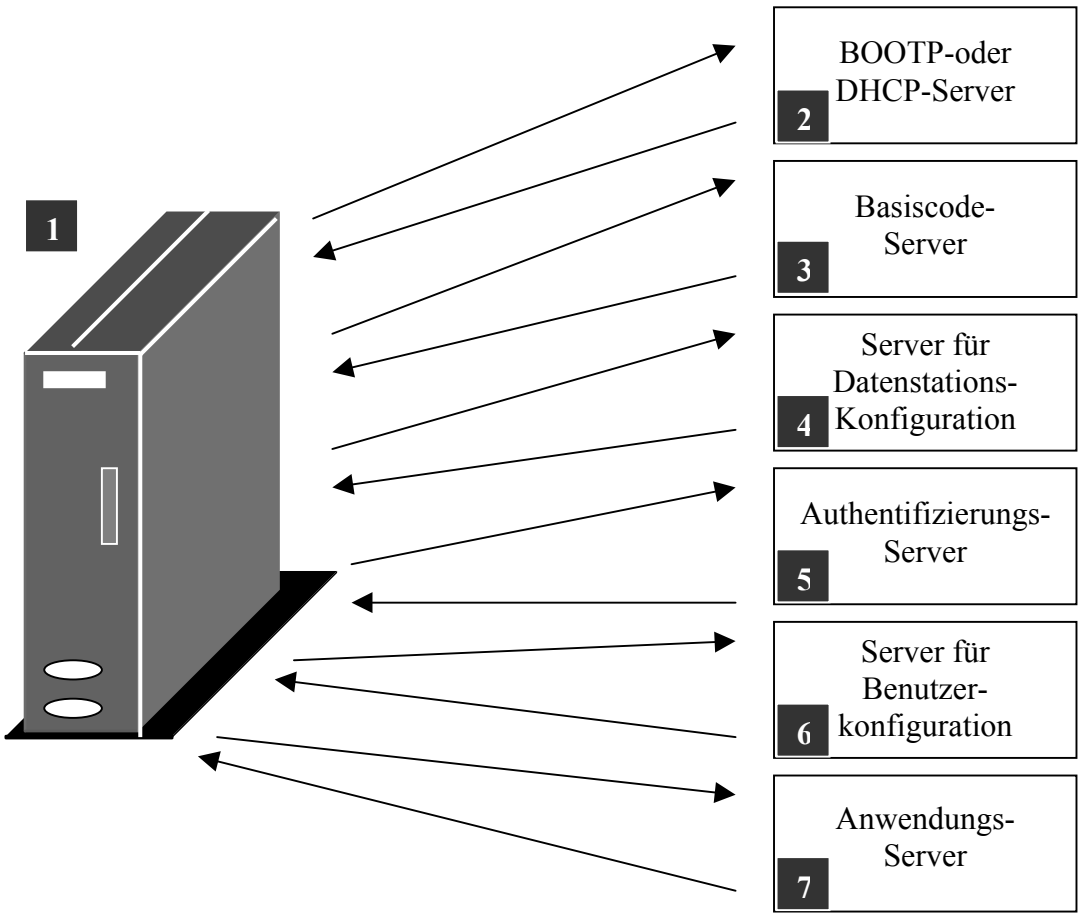


Abbildung 5.2: Reihenfolge beim Booten

Nach dem Einschalten der NS und dem Selbsttest erscheint die Nachricht NS05000. Wenn die Nachricht ”Suche nach Host-System...” erscheint, muß die Esc-Taste gedrückt werden, um das Konfigurationsprogramm zu aktivieren. Zur Zeit ist die Kennwortsteuerung nicht aktiv, d.h. man kann ohne Kennwort die Hardwarekonfiguration anzeigen und ändern. Dann erscheinen die Funktionstasten und ihre Beschreibungen [8].

Der Code des Boot-Monitors soll vor der Inbetriebnahme aktualisiert werden, um sicher zu stellen, daß die Version des Boot-Monitors auf der NS mit der Version des Boot-Monitors auf dem Server (NSM) übereinstimmt. Wenn eine Verbindung zwischen einer NS mit einem älteren Boot-Monitor-Code und einem Server, auf dem ein NSM mit seinem später hinzugefügten Boot-Monitor-Code installiert ist, hergestellt wird, aktualisiert der NSM automatisch den Boot-Monitor-Code.

Mit der Funktionstaste F2 wird nur die Hardwarekonfiguration angezeigt. Die Tasten F3, F4 und F5 sind die wichtigsten Parametereinstellungen, die vor dem Booten geprüft werden sollten.

Hinweis:
Den Benutzern der NS muß mitgeteilt werden, daß die NS während der Aktualisierung nicht ausgeschaltet werden darf. Durch das Ausschalten der NS könnte die NS physisch beschädigt werden.

Mit dem Programm IBM Network Station Manager (NSM) kann man die Konfigurationen für die NS und deren Benutzer festlegen und ändern. Über den HTTP-Server wird dem Web-Browser das Programm NSM zur Verfügung gestellt. Auf jeder NS befindet sich als Teil des Betriebssystems ein SNMP-Agent (Simple Network Management Protocol). Ein SNMP-Manager an einem zentralen Standort kann mit dem Agenten auf einer NS kommunizieren und Daten mit diesem austauschen. Diese Daten können zur Verwaltung der Netzumgebung verwendet werden. SNMP ist ein Industriestandardprotokoll für die Netzverwaltung.

Auf jeder NS kann das Konfigurationsdienstprogramm für NSs aufgerufen werden. Mit dem Konfigurationsdienstprogramm für NSs können Konfigurationseinstellungen auf einer bestimmten NS angezeigt oder festgelegt werden. Beispielsweise kann die MAC-Adresse angezeigt oder die Bildschirmauflösung der Network Station festgelegt werden.
Nach dem Laden des Basiscodes der Network Station stehen die Programme "User Services" (Benutzerservices) zur Verfügung. Benutzerservices sind Programme, die den Benutzern Hilfsprogramme zur Verwaltung der Betriebsumgebung der Network Station zur Verfügung stellen. Im folgenden sind einige der Benutzerservices aufgelistet:

- Überwachung von Nachrichten für eine bestimmte NS
- Sperren des Bildschirms (mit Kennwortsteuerung)
- Überwachung von Statistiken (z. B. verfügbarer Speicher auf einer bestimmten NS)

Weitere Informationen über die NS sind im [8] enthalten.

5.1.3 Erforderliche Informationen zu TCP/IP-Netzen

Damit die NS mit den Servern kommunizieren kann, wird ein TCP/IP-Netz benötigt. Kenntnisse über das TCP/IP-Netz erleichtern dem Benutzer die Installation und Konfiguration der NS und des Programms NSM erheblich. Um den Aufbau eines Netzes verstehen zu können, sollte man einen Netzplan erstellen. Abbildung 5.3 zeigt ein Beispiel für einen Netzplan, in dem zwei NS´s über ein einfaches lokales Netz (LAN, Local Area Network) verbunden sind.

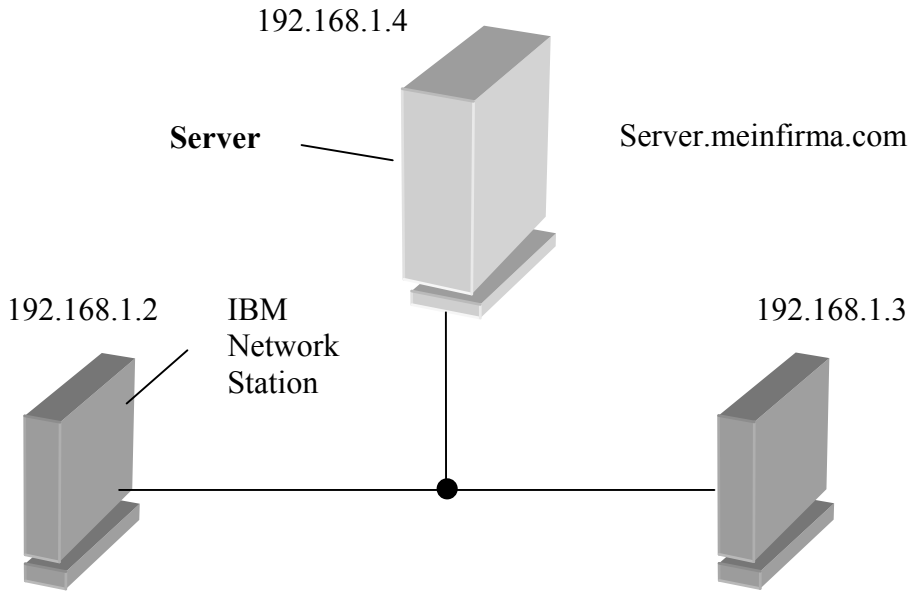


Abbildung 5.3: Verbindung von zwei NSs über ein LAN mit dem Server

5.1.4 MAC-Adressen

Jede NS verfügt über eine eindeutige Kennummer, mit deren Hilfe ermittelt werden kann, welche IP-Adresse dem NC zugeordnet ist. Die MAC-Adressen werden den NCs während der Herstellung zugeordnet und sind fest in der Maschine codiert. Man kann die fest codierte MAC-Adresse durch eine benutzerdefinierte MAC-Adresse außer Kraft setzen [8].

5.1.5 Boot-Methoden

Da eine NS über keine Platte verfügt, von der gebootet werden kann, muß sie die Informationen entweder aus ihrem NVRAM oder von einem Server abrufen. Die NS muß eine IP-Adresse für sich ermitteln. Mit Hilfe der IP-Adresse kann die NS mit anderen Hosts kommunizieren. Die Network Station muß eine der folgenden drei Methoden verwenden, um die Informationen anzufordern und zu empfangen:

- Nicht flüchtiger Arbeitsspeicher (NVRAM)
- Bootstrap Protocol (BOOTP)
- Dynamic Host Configuration Protocol (DHCP)

Jede Plattform unterstützt andere Boot-Methoden, die auf den einzelnen Plattformen verfügbar sind. In dieser Arbeit wird ausschließlich der NT-Server mit der DHCP (NC), NVRAM (NS), TFTP (NC und NS) und NFS (NS) Unterstützung verwendet.

5.1.6 DHCP

DHCP ist ebenfalls ein TCP/IP-Protokoll. DHCP bietet einem Server die Möglichkeit, IP-Adressen und Konfigurationsdaten automatisch zuzuordnen, ohne den Administrator dazu

zu zwingen, die MAC-Adressen der Computer im Netz zu notieren und verfolgen zu müssen. Mit DHCP kann jedem Host (z.B. NC oder NS) aus einem zuvor festgelegten Bereich von IP-Adressen entweder eine permanente IP-Adresse oder eine temporäre IP-Adresse zugeordnet werden. Ferner können IP-Adressen entweder statisch oder dynamisch zugeordnet werden.

Die statische Zuordnung gleicht der Art und Weise, in der BOOTP IP-Adressen zuordnet werden. Die MAC-Adressen der einzelnen NSs werden in der Konfiguration des DHCP-Servers zusammen mit einer IP-Adresse, die für die NS mit dieser MAC-Adresse reserviert ist, definiert. Wenn die NS eine Anforderung an den DHCP-Server sendet und sich durch seine MAC-Adresse identifiziert, gibt der Server die IP-Adresse, die er für diesen Client reserviert hat, zurück.

Bei der dynamischen Zuordnung von IP-Adressen identifiziert der Server die NS ebenfalls über dessen MAC-Adresse. Anstelle der Verwendung einer festen IP-Adresse ordnet der Server der NS jedoch eine beliebige Adresse aus dem verfügbaren Pool zu. Der Server weist der NS die Adresse für einen bestimmten Zeitraum zu. Die Adresse wird in den Pool zurückgestellt, wenn sie vom Client freigegeben wird oder wenn die Zuweisungsfrist abgelaufen ist. DHCP kann auch nicht aufgeführte Clients bedienen. Jeder Client kann, auch wenn seine MAC-Adresse nicht in der DHCP-Konfiguration definiert ist, eine IP-Adresse aus dem Pool der verfügbaren Adressen anfordern. Die Verwendung nicht aufgeführter Clients kann sich für Umgebungen eignen, in denen es nicht erforderlich oder gewünscht ist, MAC-Adressen zu verwalten. Während BOOTP-Server nur Anforderungen von BOOTP-Clients verarbeiten können, ist DHCP in der Lage, Anforderungen von DHCP- und BOOTP-Clients zu verarbeiten. DHCP-Server können, anders als BOOTP-Server, momentan nicht im Gebrauch befindliche IP-Adressen wiederverwenden. Ferner stellt DHCP zahlreiche Konfigurationsoptionen einschließlich benutzerdefinierter Optionen zur Verfügung. Mit diesen Optionen können viele erweiterte Netzumgebungen konfiguriert werden.

5.2 IBM Network Station Manager

Damit man die NS auf seinem Arbeitsplatz einsetzen kann, müssen die NS und die Server ständig miteinander kommunizieren. Nur über NVRAM und den lokalen Speicher kann die NS ohne Verbindung mit einem Server gebootet werden. Für weitere Funktionalität als Client muß die Verbindung herstellt werden. Als lokaler Speicher für den Basiscode (Betriebssystem) kommt eine PCMCIA Speicherkarte zum Einsatz. Da diese Speicherkarte nicht mit der NS geliefert wurde, mußte zuerst unbedingt der NSM auf dem NT-Server installiert werden.

Abbildung 5.4: Hauptanzeige des Programms NSM

Der NSM stellt aus Benutzersicht eine Desktop-Oberfläche mit einem Login am Boot-Server und einer Taskleiste mit den dem Benutzer zur Verfügung gestellten Applikationen dar. Für den Administrator bietet er eine Web-basierende Oberfläche zur Verwaltung von NSs zur Aktualisierung von Boot-Monitor-Codes und Basiscodes und zur Verwaltung der Benutzer.

Die Verwaltung der NSs erfolgt komplett im Web-Browser über den NSM. Als Administrator hat man Zugriff auf sämtliche Einstellungen, als normaler Benutzer kann man nur wenige Präferenzen für die Standardapplikationen einstellen. Die Einstellungen sind in Hardware, Systemstart, Arbeitsoberfläche, 5250 –und 3270-Sitzungen, Internet, Sprache und Benutzergruppen unterteilt (siehe Abbildung 5.4).

Alle Einstellungen lassen sich auf System-, Gruppen- und Benutzerebene setzen. Die Hardwareeinstellungen kann man zusätzlich auch für bestimmte Arbeitsplätze festlegen. Damit ist es möglich, den Benutzern z.B. nur Hardwarekonfigurationen (F2, siehe 4.2.2) anzuzeigen.

5.2.1 Voraussetzung und Installation

An dieser Stelle wurden zwei Varianten vom NSM auf dem NT-Server installiert. Zuerst wurden die Softwarekomponenten einzeln aus dem Netz heruntergeladen. Während der

Installation von Microsoft Software traten einige Probleme auf (siehe 5.4). Deshalb wurden alle Komponenten von IBM und anderer Hersteller, die auf der NSM-CD sind, installiert und konfiguriert. Bevor man jedoch mit der Installation beginnt, müssen folgende Gegebenheiten überprüft bzw. Vorleistungen erbracht werden:

- NT Server 4.0 plus Servicepack 3 mit NTFS Dateisystem und ca. 200 MByte freier Speicherplatz
- java-fähiger Web-Browser (Netscape Navigator 4.04 oder Microsoft Internet Explorer 4.0.1 als Standard-Browser)
- Netzkarte mit RJ45-Buchse
- Erstellung des Netzplans

Nach der Übernahme des NT-Servers ist der Netzplan wie folgt erstellt worden:

Feld	Beschreibung	Netzwert
TFTP- und/oder NFS-Server (IP-Adresse)	Der Server, von dem die NS ihr Betriebssystem herunterlädt.	129.69.81.52
Domänennamens-Server (IP-Adresse)		129.69.81.21
Gateway (IP-Adresse)	Die IP-Adresse des Standard-Routers, an den TCP/IP-Pakete, die nicht an das Netz adressiert sind, gesendet werden.	129.69.81.10
Teilnetzmaske	Ein Wert, mit dessen Hilfe Informationspakete von Netzeinheiten in einer Teilnetzumgebung präzise übertragen werden können.	255.255.255.0
Konfigurations-Host (IP-Adresse)	Server, von dem die NS die Konfigurationsdaten herunterlädt.	129.69.81.52
IP-Adresse der NS	Die an die NS zu vergebende Adresse, die vom Administrator erteilt werden soll.	129.69.81.54
Broadcast Adresse	Die Adresse , über die mit jedem Host im Netz kommuniziert werden kann.	129.69.81.255

Tabelle 5.2: Netzplan

Auf der NSM-CD sind folgende Dienste enthalten:

- Interner Information Server (Lotus Domino Go Webserver 4.6.1)
- IBM DHCP, DNS, TFTP –und NFS-Server
- NSM 3.0

Bei der Installation wird die lokale Festplatte nach dem erwähnten Web-Browser und nach dem IIS durchsucht. Wenn sie auf der Festplatte nicht installiert sind, wird das Installationsprogramm mit einer Fehlermeldung unterbrochen. Die Installation soll in folgender Reihenfolge durchgeführt werden:

1. Zuerst wird einer der empfohlenen Web-Browser installiert.
2. Danach wird IIS (in diesem Fall Lotus Domino Go) bereitgestellt.
3. Als nächstes wird die IBM TCP/IP-Services (NFS, TFTP und TIMED) installiert.
4. Im letzten Schritt wird die NSM-Installation ausgeführt.

Da aus Zeitgründen die DHCP-Boot-Methode nicht verwendet wird, ist für diese Komponente (DHCP) im Netzplan und für die Installationsschritte kein Eintrag der Einstellungen vorgesehen.

Die installierte Software und deren Eigenschaften sind aus Tabelle 5.3 zu entnehmen.

Komponente	**Verzeichnis**	**Erreichbar** (Verk)
Lotus Domino Go	D:\Www\	
ENetwork On Demand Server NFS-Server TFTP-Server Zeit-Server	D:\OnDemand\Server	Startmenü\Programme\eNetwork OnDemand Server\ (nur für Konfiguration)
NSM	D:\nstation\	Desktop\IBM Network Station Manager Startmenü\Programme\IBM Network Station Manager\IBM Network Station Manager

Tabelle 5.3: Installierte Software auf dem NT-Server

5.2.2 Voreinstellung der Boot- und Konfigurationsparameter

Wie vorher erwähnt (siehe 5.1.5), wird die NS -in dieser Arbeit- hauptsächlich über den NVRAM gebootet. In diesem Abschnitt werden die Einstellungen der NS anhand des Netzplans und mit Hilfe der in 5.1.2 genannten Reihenfolge beschrieben. Mit dem Einschalten der NS werden die wichtigsten Hardwareinformationen auf dem Bildschirm angezeigt. Alle Zeilen sind mit einer Nachrichtennummer versehen.

Nach dem Aufruf des Boot-Programms werden als erstes die Netzparametersfestlegungen (F3) vorgenommen. Die Parameter können der Tabelle 5.4 entnommen werden.

Konfigurationskomponente	Wert
IP-Adresse bezogen von	NVRAM
IP-Adresse der NS	129.69.81.54
Boot-Host	129.69.81.52
Gateway	129.69.81.10
Teilnetzmaske	255.255.255.0
Broadcast	129.69.81.255
Ethernet-Standard	IEEE 802.3

Tabelle 5.4: Boot- und Konfigurationsparameter (F3)

Der zweite und dritte Server (Boot-Host) dienen ebenfalls dazu, und zwar dann, wenn das Booten über den ersten bzw. zweiten fehlschlagen sollte. Das gleiche gilt für den Konfigurations-Host. Weil es sich hier um denselben Server für das Booten und Konfigurieren handelt, kann der Wert 0.0.0.0 oder die IP-Adresse des ersten Servers eingegeben werden.

Als nächstes müssen die richtigen Pfade eingeben werden, damit die NS die Boot- und Konfigurationsdateien lokalisieren kann. Das geschieht mit dem Drücken der F4-Taste. Diese Boot-Parameter sind ausführlich in Tabelle 5.5 erläutert.

Boot-Parameter	Beschreibung	Wert
Boot-Datei	Die Datei, die das Betriebssystem für die NS enthält.	kernel
TFTP-Boot-Verzeichnis	Der Pfad, den die NS für den Zugriff auf die Boot-Datei auf dem Boot-Server verwendet.	/nstation/prodbase/
NFS-Boot-Verzeichnis	Der Pfad, den die NS für den Zugriff auf die Boot-Datei auf dem Boot-Server verwendet.	/netstation/prodbase/
TFTP (Boot-Host)	Protokollreihenfolge	2
NFS (Boot-Host)	Protokollreihenfolge	1
Lokal	Booten von einer FLASH-Karte	Inaktiviert

Tabelle 5.5: Boot-Datei, Pfad und Protokolle für die NVRAM-Boot-Methode (F4)

Folglich müssen die Parameter für die Konfiguration der NS festgelegt werden. Dazu sollte die F5-Taste gedrückt werden. Um diese Konfigurationsdatei festzulegen, kann die Tabelle 5.6 verwendet werden.

Konfigurationsdatei	Beschreibung	Wert
Konfigurationsdatei	Der Name der Datei, die die Konfigurationsdaten der NS enthält.	standard.nsm
Konfigurationsverzeichnis	Der Pfadname, den der Konfigurations-Host für die Ermittlung der Konfigurationsdatei verwendet.	/netstation/prodbase/configs/
1. Protokoll	Konfiguration-Server-Protokoll	NFS
2. Protokoll	Konfiguration-Server-Protokoll	TFTP

Tabelle 5.6: Konfigurationsdatei für die NVRAM-Boot-Methode (F5)

Hinweis:
Für den Bildschirm muß über F6 die vom Monitor unterstützte Auflösung richtig eingestellt werden. Die automatische Einstellung soll auch vermieden werden, da der Monitor im ersten Moment ständig aus- und eingeschaltet wird. Diese wechselnde Umschaltung kann den Monitor beschädigen.

Die Auflösungen für Boot-Monitor und Server sind mit den Nachrichtennummern NS070 und NS080 in Verbindung mit anderen Hardwareinformationen (siehe oben) auf dem Monitor zu sehen.

5.2.3 Arbeitsumgebung auf der NS

Nachdem die in 4.2.2 beschriebenen Dienste durch den Boot-Monitor und teilweise durch das entpackte Betriebssystem (OS-9000) eins nach dem anderen durchgeführt worden sind, erscheint ein Anmeldefenster. Wenn kein neuer Benutzer für den NS-Arbeitsplatz definiert wird, kann sich der Administrator unter gleichem Benutzernamen und Kennwort beim Server einloggen. Diese Daten werden bei der Installation von NSM automatisch abgefragt. Ansonsten kann jeder, der als Benutzer vom Administrator definiert ist, auf den NSM zugreifen. Es ist auch möglich, daß der Administrator keine Berechtigung erteilt. Auf jeden Fall verweigert das Installationsprogramm dann eine Installation des NSMs.

Nach dem Einloggen wird die Verbindung mit dem Identifikationsüberprüfungsserver hergestellt. Nach kurzer Zeit wird der Standard-Desktop mit einer Menüleiste aufgebaut, „Standard“ deshalb, weil der NSM zuerst die Standard-Konfigurationsdatei für jeden Benutzer verarbeitet. Danach kann der Benutzer seinen Arbeitsplatz nach seinen Wünschen über den NC- Navigator ändern. Dazu muß er den NC-Navigator starten und den NSM aufrufen. Jeder Benutzer kann nur auf bestimmte Einstellungsdateien zugreifen.
Auf der Menüleiste sind außer den standardmäßig verfügbaren Anwendungsknöpfen (5250, 3270 und NC Navigator-Browser) auch selbst entwickelte Java-Applets und –Anwendungen zu sehen. Sie werden hier aufgeführt:

Applet:
DigitalClock
Anwendung:
QuadraticGraph, SerialDemo, Sysproperties, SimpleRead und SimpleWrite

5.3 Untersuchung der Java Virtuelle Maschine (JVM)

Über den NSM können Java-Applets und –Anwendungen automatisch oder durch Menüpunkte (als Knöpfe in der Menüleiste) durchgeführt werden. Es muß darauf hingewiesen werden, daß jeweils nur eine Anwendung auf der NS ausgeführt werden kann. Der Benutzer muß sich entweder für Applet oder für Anwendung entscheiden. Allerdings ist es möglich, mehrere Java-Applets gleichzeitig auszuführen.
Da keine Dokumentation über die JVM auf der NS verfügbar ist, mußte in dieser Arbeit die JVM Schritt für Schritt untersucht werden. Es gab kein Hinweis auf eine Klasse oder ein Paket, um einen Port zu unterstützen. Weiterhin wurde eine praktische Untersuchung vorgenommen. Zur Überprüfung der JVM wurden die oben aufgeführten Programme auf der NS ausgeführt.
Die Programme DigitalClock, QuadraticGraph und Sysproperties können jederzeit einwandfrei an der NS ausgeführt werden. Die drei Anwendungen jedoch, die eine einfache Funktionalität der seriellen Schnittstelle ermöglichen, können nicht an der JVM ausgeführt werden. Der ganze Ablauf, vom Starten bis zum Beenden der JVM, kann über die NS User Services (Console) beobachtet werden.

Auf der Console werden die vom System durchgeführten Aktionen ununterbrochen angezeigt. Die Console wird durch Drücken der Tastenkombination Umschalttaste, Alt und Pos1 aufgerufen. Außerdem kann der Benutzer die Speicherbelegung durch den JVM ständig über Menü (Console), Statistics und Show Memory überprüfen.

Die Eigenschaften der NS, die durch Ausführung des Programms Sysproperties auf der Console angezeigt wurden, sind in Abbildung 5.5 zusammengeführt.

Nach der Ausführung der Anwendungen SerialDemo, SimpleRead und SimpleWrite wurde die JVM gestartet und wegen fehlender Klassen sogleich auch beendet. Als Beispiele werden die Zeilen, die nach dem Start der JVM auf der Console angezeigt wurden, unten aufgelistet.

Java.lang.NoClassDefFoundError: javax/comm/CommPoreIdentifier
at
at SerialDemo$ConfigurationPanel.<init>(SerialDemo.java)
at SerialDemo.<init>(SerialDemo.java)
at SerialDemo.main(SerialDemo.java)

```
--listung properties
user.language=de
java.home=/netstation/prodbase/Java
file.encoding.pkg=sun.io
user.emailname=irani
file.separator=/
line.separator=

user.region=DE
file.encoding=8859_1
java.vendor=IBM Corporation
user.timezone=GMT
user.name=irani
os.arch=PowerPC
os.name=NCDWare
java.vendor.url=http://www.ibm.com(
user.dir=/netstation/homebase/users/IRANI
java.class.path=/netstation/homebase/users/IRANI/:/ne...
java.class.version=45.3
os.version=4.6
path.separator=:
user.home=/netstation/homebase/users/IRANI
```

Abbildung 5.5: Ausgabe des Programms Sysproperties

Die Untersuchung der JVM wurde mit dem Entpacken von Entwickungsklassen des NSMs und der JVM weitergeführt. Die zusammengepackten Klassen **classes.zip, nwshacl.zip und nwspackg.zip** sind im Verzeichnis D:\nstation\prodbase\java\ zu finden.

Die Ergebnisse der Untersuchung wurden immer im Teamgespräch erläutert. Als nächstes wurde versucht, Kontakt mit dem Softwareentwickler der NS bei IBM aufzunehmen. Nach langer Zeit wurde mitgeteilt, daß tatsächlich kein javax.comm-Paket in der JVM für NS portiert wurde. Man sei dabei, ein javax.comm-Paket speziell für diese NS (Serie 1000; 8362-A53) zu entwickeln. Dieses wird voraussichtlich im Frühjahr 1999 auf den Markt kommen.

5.4 Problematik und Fehlerbehebung

Dieser Abschnitt enthält Informationen über die Vorbereitung von Software und Hardware des Servers und der NS. Sie können den Benutzer bei der Behebung von Fehlern und bei zukünftigen Installationen unterstützen.

5.4.1 Softwareproblem

- Da am Anfang der NT-Server abgekoppelt (nicht Intranet) vom Netz konfiguriert wurde, war es nicht möglich, den MS Internet Explorer 4.0 zu installieren, weil es sich um einen Aktiv Setup handelt. Nach Absprache mit dem Administrator wurde das Problem bewältigt.
- Das zweite Problem tauchte bei der Durchführung und beim Zusammenführen der *.EXE-Dateien vom MS Internet Information Server Pack 4.0 auf. Es wurde mehrmals auf verschiedenem Wege versucht, den Server zu installieren. Aus Zeitgründen wurden diesbezüglich keine weiteren Versuche mehr durchgeführt.
- Daher wurde von der NSM Software des Herstellers IBM Gebrauch gemacht. Da Network Computing ein neues Gebiet ist, sind auch nicht aufgetretene Probleme und Fehler gut dokumentiert. Es kommen immer mehr Fehler hinzu, die nicht direkt mit dem Thema zu tun haben.
- Die ersten Installationen schlugen stets fehl. Grund dafür war die Abkopplung vom Netz (Intranet). Der NSM verwendet die Daten vom Netzplan wie z.B. die IP-Adresse vom Router. Deshalb soll die Verbindung mit dem im Netzplan eingetragenen Server sichergestellt werden.
- Die älteren CD-ROMs haben Probleme mit dem Lesen der Sicherheitskopie (CD). Wenn die Original-CD verwendet wird, kommt selten ein Fehler vor.
- Bei der NSM-Installation wird nach den vorhandenen Diensten gesucht. Bei der Frage nach dem DHCP-Server soll dieser nicht ausgewählt werden, weil zwischen manchen Netzkartentreibern und IBM NDIS Konflikte vorkommen. In dieser Arbeit wurde keine DHCP Boot-Methode verwendet.
- Es kommt bei der Installation der TCP/IP-Software (On Demand Server) manchmal zu Problemen. Dann ist der NSM nicht in der Lage, die NFS- und TFTP-Server zu konfigurieren. Es gibt zwei Möglichkeiten, nämlich entweder die Konfiguration manuell zu übernehmen oder zuerst diese Dienste zu installieren und danach den NSM. Die zweite Methode ist sicherer und einfacher.

Der davor genannte Installationsfehler zeigt sich eher beim NFS-Server, da dieser Server, der bei NT als Dienst definiert wird, nicht gestartet werden kann. Das ist sehr wichtig, da nach dem Booten nur dieser Server für die Datenübertragung in Frage kommt.

5.4.2 Hardwareproblem

Die erste erfolgreiche Installation wurde mit einer in den NT-Server eingebauten nicht automatischen Umstellung erreicht. Diese Netzkarte hat zwei Anschlüsse: BNC und RJ-45. Man muß sich entscheiden, welcher von beiden Anschlüssen für die Verbindung in Frage kommt. Die Umstellung von dieser Karte wird softwaremäßig unter DOS vorgenommen. Nach der NSM-Installation muß der NT-Server nicht unbedingt mit dem Intranet verbunden werden. Nach einigen Tagen konnte noch immer keine Verbindung hergestellt

werden. Deshalb mußte der NSM und die notwendige Software nochmals installiert werden. Als Verbindungskabel wurde ein RJ-45 Kreuzkabel verwendet.

Zur Arbeitserleichterung wurde die alte Netzkarte gegen eine mit automatischer Umstellungsfähigkeit ausgetauscht. Diese Karte hat drei Anschlüsse: BNC, AUI und RJ-45. Je nachdem, welche Stecker angeschlossen sind, werden die Ein- und Ausgaben zu diesem Anschluß umgeleitet. Obwohl die Software unter Berücksichtigung der bis dahin aufgetretenen Fehler (oder Probleme) mehrmals erfolgreich installiert wurde, konnte keine Verbindung zwischen dem NT-Server und der NS hergestellt werden. Mit der Änderung der Blockgröße konnte man teilweise die Verbindung zum TFTP-Server aufbauen. Das aber reichte nicht aus, um den Basiscode zur NS zu transportieren. Nach mehreren Wochen konnte das Problem lokalisiert und gelöst werden.

Man muß die Tatsache beachten, daß die Kreuzkabel nicht einwandfrei in einem Netz mit Netzkarte, die über eine automatische Anschlußerkennung verfügt, eingesetzt werden können. In einem solchen Fall wird eine Verbindung über einen HUB (Sternverteiler) dringend empfohlen.

6 Sensorik und Aktorik in einem Fließprozeß

Ein am Institut für Automatisierungs- und Softwaretechnik stehender industrieller Kaffeeautomat der Firma WMF (mit der Bezeichnung etamat ECF 705500) soll zur Demonstration einer Betriebsüberwachung und Fernsteuerung umgebaut werden, damit man ihn über das Netz überwachen und bedienen kann.

Dabei werden durch eine Vielzahl von Sensoren, Betriebsgrößen wie Temperatur, Füllstände, Durchflußmengen, elektrische Spannungen erfaßt und visualisiert. Diese Daten werden zur Zeit durch im Automat und in nebenliegender Blackbox eingebaute Karten erfaßt und durch eine unter Turbo-C entwickelte Anwendung (*Kaffee.exe*) visualisiert.

Es ist erwünscht, den Automat via Netz zu bedienen. Als Programmiersprache wird Java eingesetzt, damit man die Anwendungen auf unterschiedliche Plattformen ausführen kann. Die verwendete Hardware, Blackbox, Datenerfassungskarten und zusätzlich Aktoren sollen durch einen leistungsfähigen Rechner und ein kompakte Lösung ersetzt werden.

Ziel der Diplomarbeit war es, zuerst die richtigen Karten unter den geforderte Aspekte zu suchen. Danach sollten die Signale gemeinsam auf einer Karte bearbeitet und weitergeleitet werden. Dazu gehören die Temperaturverstärkern, die Pegelanpassung für Überwachungssignale, Umschaltung für lokale und ferne Steuerung und Zuführung von Aktorsignalen. Um die Sensor- und Aktordaten im PC weiterzuverarbeiten, sollten die Signale über zwei Schnittstellen am Kaffeeautomat zur Verfügung gestellt sein.

6.1 Technische Realisierung

Aufgabe der Diplomarbeit war es, eine Lösung zu konzipieren, die es ermöglicht, die Prozeßdaten über eine eingebaute Schnittstelle mit wenige Aufwand und Zusatzgeräte zu übertragen.

Im Rahmen dieser Arbeit wurden die Schaltungen für die Pegelanpassung der Aggregatzustände (Sensorik), Umschaltung der Auswahltasten (Aktorik) und die Temperaturmessung mit Thermoelement entworfen und realisiert. Folgend werden die Grundlagen für die Temperaturerfassung, Pegelanpassung und elektronische Umschalter erläutert. Anschließend werden die realisierten Schaltungen kurz beschrieben.

6.2 Temperaturmessung

Die Temperaturmessung mit Thermoelement ist eine seit langem bewährte Methode der Temperaturmessung. Ihre Vorteile liegen in der hohen Genauigkeit und dem sehr großen Temperaturbereich von über 2000°C. Der einfache Aufbau aus zwei verschiedenen, an der Spitze verschweißten Metalldrähten macht sie obendrein preiswert. Thermoelemente gibt es in den verschiedensten Ausführungen für alle nur erdenklichen Anwendungen. Von in Stahlhülsen eingeschweißten Typen zur Temperaturmessung in Reaktoren bis zu haarfeinen Sonden, mit denen die Temperatur von Eizellen gemessen werden kann. Ein weiterer Vorteil der Thermoelemente ist die geringe Empfindlichkeit gegenüber äußeren Einflüssen wie Hitze, Kälte, Druck oder Vibration. Als problematisch erweist sich dagegen aufgrund der kleinen Pegel die Signalaufbereitung.

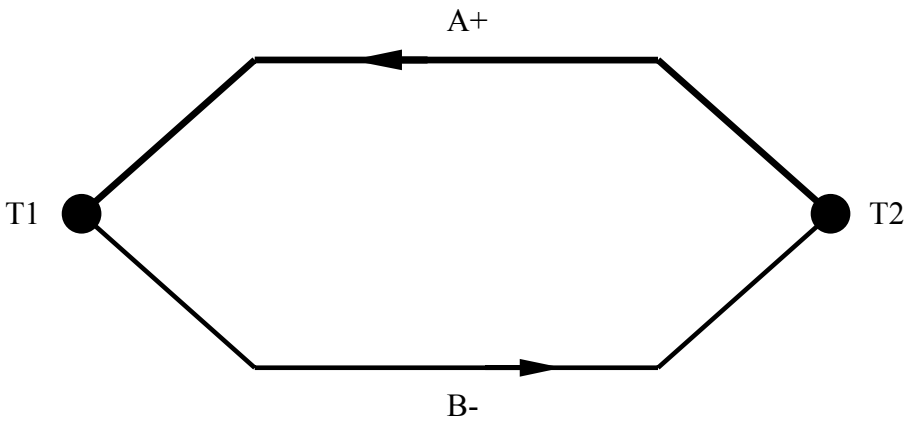


Abbildung 6.1: geschlossenes Thermopaar

Die Temperaturmessung mit Thermoelement ergibt sich aus dem Seebeck-Effekt. Dieser besagt, wenn zwei Leiter unterschiedlicher Materialien wie in Abbildung 6.1 verbunden sind, fließt ein Strom, wenn sich die beiden Verbindungspunkte auf verschiedenem Temperaturniveau befinden. Dieser Strom ist abhängig vom Temperaturunterschied der beiden Verbindungsstellen. Sind beide auf gleichem Temperaturniveau, fließt kein Strom. Die beiden Leiter bilden das Thermopaar.

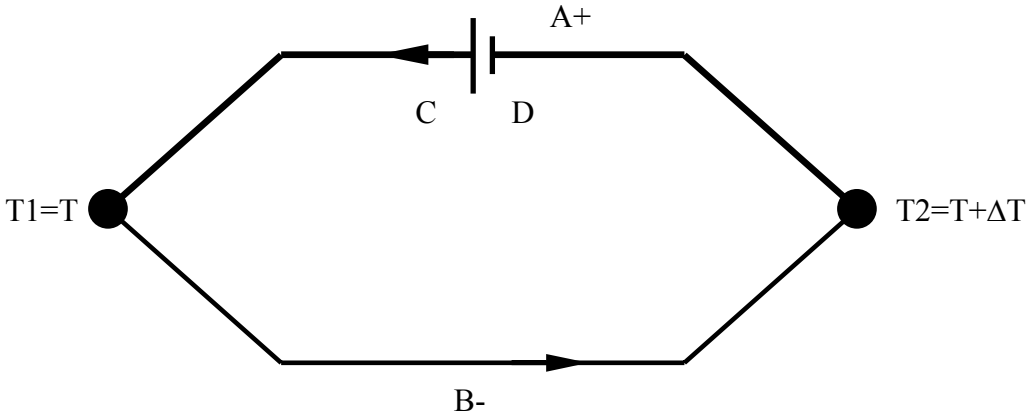


Abbildung 6.2: offenes Thermoelement

Es ist leicht einzusehen, daß bei einer Anordnung wie in Abbildung 6.2 kein Strom mehr fließt, sondern eine von den Temperaturen abhängigen Spannung zwischen den Punkten C und D abfällt. Wird nun in den Punkten C und D beispielsweise eine Verstärkerschaltung angelötet, so ergeben sich zwei weitere Thermopaare, die zu einer Verfälschung führen könnten. Sorgt man allerdings dafür, das sich die Punkte C und D auf der gleichen Temperatur befinden, so heben sich die Einflüsse der beiden Übergänge auf, da sie entgegengesetzt gepolt und gleich groß sind.

Die Beziehung zwischen Thermospannung und Themperaturdifferenz ist im allgemeinen nicht linear, sie ist eine Gleichung dritten Grades. In den üblichen Temperaturbereichen technischer Messungen genügt jedoch meistens die Annahme einer quadratischen Abhängigkeit als Approximation.

$$\Delta U = a + b\Delta T + c\Delta T$$

In dieser Gleichung sind a, b und c materialabhängige Konstanten. Sie werden durch Eichen an Fixpunkten bestimmt. Häufig reicht aber bei vielen Thermopaaren eine Linearisierung aus, wenn die zu messende Temperaturspanne nicht zu groß ist. Die Ungenauigkeiten sind bei geeigneter Wahl von Temperaturspanne und Thermopaar zu vernachlässigen.

$$\Delta U = k\Delta T \qquad\qquad k = [mV/K]$$

Die darin auftretende Konstante K ist vom Temperaturniveau abhängig. Meistens ist die Vergleichstemperatur aber nicht bei 0°C, sondern liegt z.B. im Bereich 20°C bis 60°C. Noch problematischer ist eine nicht konstante Vergleichstemperatur. In diesem Fall muß der gemessenen Thermospannung U_m eine Korrekturspannung U_k hinzugefügt werden, die der Temperaturabweichung der Vergleichtemperatur von 0°C entspricht.

$$\Delta U = \Delta U_m + \Delta U_k$$

Im Rahmen dieser Arbeit wird die Methode der Temperaturdifferenzmessung bei variablen Außentemperatur realisiert. Unter Berücksichtigung verschiedener Gesichtspunkte wie Linearität des Thermospannungsverlaufs, günstiges dynamisches Verhalten, Korrosion im Temperaturbereich usw. wurde das NiCr/Ni Thermoelement ausgewählt. Das ist ein unedles Thermopaar, das mit 0,04 mV/°C einen relativ hohen mittleren Temperaturkoeffizienten hat. Die Linearität ist besser als bei den meisten anderen gängigen Thermoelementen und eine Linearisierung muß nur bei der Erfassung eines großen Temperaturbereichs durchgeführt werden. Das verwendete Element ist ein koaxial abgeschirmtes, von einem Metallrohr ummanteltes Thermoelement.

6.2.1 Erfassung der Temperatur

Da die Thermospannungen lediglich im µ-Volt Bereich liegen, ist die Signalverarbeitung nicht unproblematisch. Das Meßsignal muß verstärkt werden, und um eine auf 0°C bezogene Spannung zu erhalten, eine Nullpunktverschiebung durchgeführt werden. Es empfiehlt sich eine existierende integrierte Schaltung, den AD595 CQ von Analog Devices, für die erste Vorverstärkung einzusetzen. Der IC (Integrierte Circuit) verstärkt die Thermospannung von ca. 0,04 mV/°C auf 10 mV/°C, Des weiteren wird gleich eine Nullpunktkompensation durchgeführt. D.h. die ausgegebenen Spannungen beziehen sich auf 0°C, somit wird keine Referenztemperatur mehr benötigt. Die Funktionsweise des AD595 CQ wird später genauer beschrieben.

Der AD595 CQ ist für Thermoelemente vom Typ K(NiCr/Ni) dimensioniert. Für Thermoelement Typ J muß ein äquivalenter IC von Analog Devices, der AD594 CQ, verwendet werden. Die weitere Verstärkung kann nun mit einfachen Mitteln geschehen, da die Pegel 10mV/°C nicht mehr störanfällig sind.

6.2.2 Die Temperaturverstärker

Die am NiCr/Ni abfallende Spannung soll so verstärkt werden, daß die Dynamik des A/D Wandlers der ein gebauten Karte voll ausgenutzt werden kann. Das bedeutet, daß die zu erwartenden Signalspannungen, des interessierenden Temperaturbereichs, so gespreizt und verschoben werden müssen, daß sie auf den vollen Spannungsbereich des AD Wandlerseingang abgebildet werden. Obwohl die ausgewählte Karte eine Spanne von bis zu ±10V unterstützt, wurde versucht die Realisierung möglichst allgemein zu halten, da es sich hier um eine häufig benötigte Schaltung handelt. Um das zu erreichen, wird zusätzlich zu dem variablen Verstärkungsfaktor eine Einrichtung zur Nullpotentialverschiebung benötigt. Die Ausgangsspannung des AD595 CQ wird so verstärkt und verschoben, daß der interessierende Temperaturbereich , z.B. 60°C bis 100°C, zur Ausnutzung der vollen Dynamik eines A/D Wandlers auf den Spannungsbereich 0 Volt bis 5 Volt abgebildet wird. Auf diese Art wird der Quantisierungsfehler bei der Digitalisierung so gering wie möglich gehalten.

Die Schaltung besteht aus vier hintereinander geschalteten Stufen.

1. Monolithic Themopaarverstärker mit Eispunktkompensation
2. Spannungsfolger
3. Verstärker mit variablem Verstärkungsfaktor
4. Schaltung zur Nullpotentialverschiebung (-12 V...+12 V)

Weil die verwendete Ein/Ausgabe Karte bipolare Spannungen unterstützt, wird die letzte Stufe, d.h. Nullpotentialverschiebung, nicht mehr in Betrieb genommen. Die ausgelieferte

Spannung vom Vorverstärker wird durch den invertierenden Verstärker um V=8 verstärkt. Auf einer Platine (siehe Anhang) wurden vier (A, B, C und D) dieser Schaltungen auf der linken Seite nebeneinander realisiert. Die Ausgänge der Schaltungen sind auf die Schnittstelle *Temperatur* (D-Sub 9polig) auf der Rückseite nach außen geführt. Die Pinbelegung der Schnittstelle ist der Ausarbeitung beigefügt.

Die realisierte Schaltung kann auch als Kleinsignalverstärker verwendet werden, wenn die Eingänge nach den Thermoelementverstärker (AD595) angeschlossen werden (siehe Anhang). In diesem Fall kann entweder der Vorverstärker ausgesteckt oder nicht eingebaut werden. Zur Eichung der Temperaturverstärkern wird der Thermoelement-Simulator mit dem Typ 4521 NiCr-Ni DIN von der Firma burster präzisionsmeßtechnik eingesetzt. Zu jedem Kanal des Verstärkers gehören zwei Potentiometer, an denen Verstärkungsfaktor und Nullpotentialverschiebung eingestellt werden können.
Das Layout wurde mit dem CAD Programm EAGLE 3.55 erstellt. Es ist auf der Diskette unter „TePaUs.brd“ gespeichert.

6.2.3 AD 595 CQ von Analog Devices

Der AD 595 CQ von Analog Divices ist so konzipiert, daß ohne zusätzliche äußere Beschaltung, bei Anschluß eines NiCr/Ni Thermoelements, am Ausgang eine Spannung von 10 mV/°C abgegriffen werden kann. Da Chip-Intern bereits eine Nullpunktkompensation duchgeführt wird, ist bei dieser Meßmethode keine Referenztemperatur, wie ein Eisbad, mehr nötig. Um das zu erreichen, sind im IC folgende Baugruppen realisiert:

- Ein sehr guter driftarmer Verstärker
- Ein Temperaturfühler zur Messung der Referenztemperatur
- Eine vom Temperaturfühler gesteuerte Spannungsquelle zum Nullpunktabgleich
- Ein Summierverstärker um Meßsignal und Referenzspannung zu addieren
- Ein Endverstärker um auf die 10 mV/°C zu kommen

Betrachtet man die Genauigkeit des AD595 CQ, so muß noch einmal darauf hingewiesen werden, daß keine Linearisierung durchgeführt wird. Ungenauigkeiten des Thermoelemente werden mit verstärkt. Da die IC-Temperatur im Bereich 0°C bis 50°C variieren darf, entsteht ein Driftfehler, der in Abbildung 6.3 in seinen Maximalgrenzen dargestellt ist. Für die Aufgabe im Rahmen der Diplomarbeit ist dieser Driftfehler nicht von Belange, da in den Räumlichkeiten nicht mit Temperaturschwankungen zu rechnen ist. Ein Fehler infolge Eigenerwärmung ist aufgrund geringem Stromverbrauch und kleinem thermischen Widerstand der IC Gehäuses zu vernachlässigen.

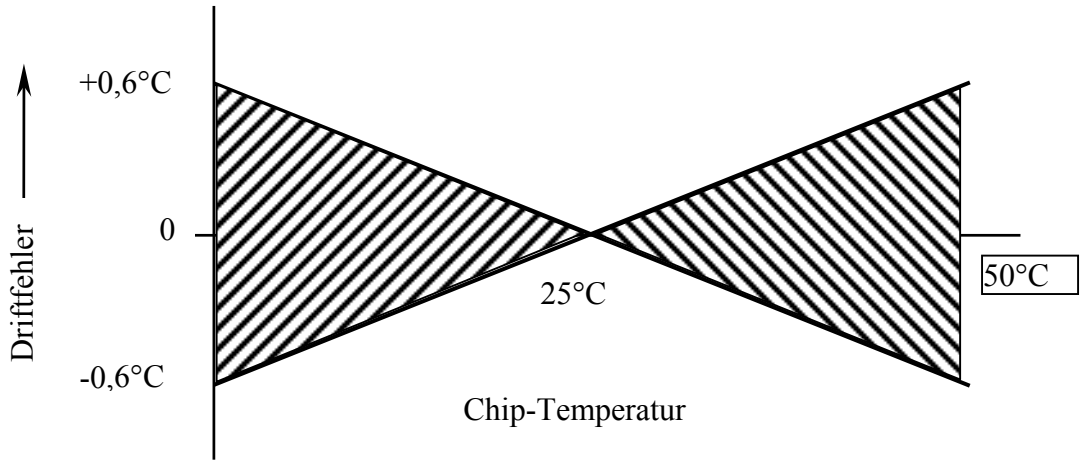


Abbildung 6.3: Driftfehler des AD 595 CQ

Im Anhang ist die gesamte Verstärkerschaltung und die Grundschaltung des AD 595 CQ für Temperatur von 0°C bis 300°C aufgeführt. Sollen größere oder negative Temperauren gemessen werden, so ist der AD 595 CQ anders zu beschalten. Dies ist aus einer Beschreibung und den Datenblättern im Anhang zu entnehmen.

6.3 Das Puffern von Sensorsignalen

Um die Ventile, Kaffeemühle und andere Aggregate in Gang zu setzen, werden in diesem Automaten Transistoren als Schalter verwendet. Die ankommenden Signale wie z.B. durch den Endschalter erzeugt und ausgehende Signale, die durch den Mikrocontroller generiert werden, werden durch vorgesehene Bus-Leitungstreiber (TTL-Baustein) verarbeitet und zum Mikrocontroller und zu den Schalttransistoren weitergeleitet. Damit die Amplitude der Signale bei der Übertragung groß genug sind, müssen sie durch einen Treiber verstärkt werden. Das verwendete TTL-Baustein hat die Bezeichnung SN74LS465N. Dieser Baustein enthält acht nicht invertierende Puffer mit gemeinsamer Freigabe und Tristate-Ausgänge (siehe Anhang). Diese vier Treiber befinden sich ebenfalls auf der Platine, deren Ausgänge auf der Schnittstelle Sensorik auf der Rückseite geführt sind. Die Schnittstelle ist eine D-Sub (37polig) Steckverbindung. Die Stiftbelegung ist aus dem Anhang zu entnehmen.

6.4 Der Aktorumschalter

Da eine Fernsteuerung des Kaffeeautomaten erwünscht ist, sollen die lokalen Bedienungstastern deaktiviert sein. Die Elektronikkarte des Automaten bietet keine Möglichkeit, um ein Umschalten zwischen lokalen Tasten (Schließer) und elektronischen Fernschaltern zu realisieren. Deshalb sind zusätzlich drei 2-zu-1-Datenselektor/Multiplexer ausgewählt und ebenfalls auf der Platine hinzugefügt worden. Sie befinden auf der rechten Seite (Umschalter). Über einen Pfostenstecker (26polig) werden die Bedienungstasten und Eingänge auf der Elektronikkarte zu den Bausteinen geführt. Um die Verbindung zwischen den Bedienungstastern und den auf der Elektronikkarte gelöteten Bausteinen zu trennen,

sollen entweder Leiterbahnen geschnitten oder die Tasten umgebaut werden. Weil eine Beschädigung der Elektronikplatte soweit wie möglichst vermieden werden soll, wurden die Taster umgebaut. Zuerst wurden alle Taster entlötet, danach sind die zwei Pins der Taster abgebogen, damit sie keine Verbindung zur Elektronikkarte haben. Durch die entstandenen Löcher sind die Leitungen vom Pfostenstecker zur Tastenpins weitergeführt und zusammengelötet. Die Eingänge der Fernschaltern sind auf die D-Sub Schnittstelle (15polig) auf der Rückseite (Aktorik) geführt. Die Pinelegung der Schnittstelle ist im Anhang aufgeführt.

7 Zusammenfassung

Die Spezifikation der zugehörigen Hardwarekomponenten wurde in dieser Arbeit ebenso vorgestellt wie das Konzept für die Installation des Network Station Managers (NSM). Darüber hinaus wurde der Einsatz der unter Java entwickelten Anwendungen in Verbindung mit dem Hardwarezugriff auf der NS getestet und dadurch das Zusammenspiel und die Richtigkeit des Ansatzes bestätigt.

In einer weiterführenden Arbeit sollte eine Erweiterung der JVM um das *javax.comm*-Paket vorgenommen werden, mit dem die Komponenten für den Hardwarezugriff wie z.B. die serielle und die parallele Schnittstelle zur Verfügung gestellt werden. Danach sollen die Klassen und andere Komponenten entworfen und entwickelt werden. Es bleibt abzuwarten, wann das erste Kommunikationspaket (javax.comm) von IBM zur Verfügung gestellt wird oder diese Umgebung unter JNI selber realisiert wird. Dazu sind die Dokumentation des Betriebssystems und eventuell spezielle Hardware nötig.

Zukünftig wird die JVM um solche Pakete (javax.comm) erweitert. Das auf Java basierende und weiterentwickelte Betriebssystem JavaOS-2.0 wird an dieses Modell (8362-A53) angepaßt. Infolgedessen ist es dann für den Entwickler einfacher, unter Java auf die Hardware Zugriff zu haben, denn in JavaOS ist dieses Paket enthalten.

Im Rahmen dieser Arbeit wurde ein Konzept für den Zugriff auf eine serielle Schnittstelle, die Konfiguration des NCs, und die Realisierung des javax.comm-Paketes auf der JVM der NS vorgestellt. Als Hardwareplattform standen eine NC Referenz Plattform und eine NS mit dem Echtzeitbetriebssystem OS-9000 zur Verfügung. Die NS ist eine Weiterentwicklung des NCs. Beide Plattformen sind mit PowerPC-Architektur Prozessoren versehen.

Es wurden mehrere Konzepte untersucht, wie das Ablaufverhalten der JVM und des JavaOS an die Anforderungen der Applikation und der Applets optimal verwendet werden kann. Dadurch kann sich der Entwickler von Java-Applikationen auf die eigentliche Anwendung konzentrieren und muß sich weniger um die implementierungsspezifischen Details von Hardware kümmern. Zur Demonstration der Kommunikation zwischen zwei Rechnern und der Programmerstellung unter Java wurde eine Beispielapplikation erstellt, die den Nachrichtenaustausch mit einem anderen Computer über Nullmodemkabel vorführt.

Auf der Grundlage des in dieser Arbeit entwickelten Konzepts können Java-Applikationen, welche auf eine serielle Schnittstelle zugreifen, unter dem Import des *javax.comm*-Paketes erstellt werden. Die prototypische Implementierung realisiert allerdings nur die Grund-

funktionalitäten der seriellen Schnittstelle. Für eine weitergehende Implementierung ist durch diese Arbeit und die darin untersuchten Plattformen eine ausreichende Basis geschaffen worden.

Die JVM und das darin integrierte Kommunikationspaket erlauben es, Hardwarezugriffe über serielle und parallele Anschlüsse zum Datenaustausch durchzuführen. Da Java dem Entwickler keinen direkten Zugriff auf die Hardware gewährt, muß die notwendige Komponente entwickelt oder erworben werden.
Offen bleibt die Frage, ob sich die in dieser Arbeit entwickelte und spezifizierte Konzeption für die Spezifikation von Hardwarekomponenten bewähren kann und auf andere Komponenten übertragbar ist.

Der Konzeption folgte anschließend die Implementierung einer Schnittstelle am Kaffeeautomaten des Typs WMF etamat ECF 750500. Dazu wurden zwei PC-Steckkarten ausgesucht. Eine (PET48DIO) ist für die Sensoren und die Zweite (PCL 818L) ist für die analoge Temperaturerfassung und Aktorik vorgesehen. Um die Temperaturerfassung, die Pufferung von Sensorsignalen für Übertragung über Kabel und Umschaltung zwischen lokale und ferne Bedienung zu realisieren, ist ein gemeinsames Layout entworfen worden. Am Ende wurden Erfassungskarte und PC-Steckkarte auf Funktionalität geprüft.

Persönliche Erfahrungen

Bei der Bearbeitung dieser Diplomarbeit konnten einerseits die Programmierkenntnisse vertieft werden. Dabei wurden Kenntnisse in der Smart Card, OPC-Richtlinien, JavaOS, Softwareinstallation und Vernetzung von NC, NT-Server und NS mit den Werkzeugen für Administration erworben. Auch die Vertiefung der Programmierfertigkeiten in Java ist in diesem Zusammenhang zu nennen. Außerdem mußten auch die Konzepte der Network Computing, dessen Hardware und PowerPC-Prozessor, serielle Schnittstellen und Client/Server Struktur untersucht werden, so daß während der Durchführung der Diplomarbeit interessante Einblicke in dessen Syntax und Architekturen gewonnen werden konnten.

Nicht zuletzt konnte durch den Einsatz des IAS-Vorgehensmodells die Diplomarbeit als Projektarbeit aufgefaßt und durchgeführt werden. Da für die Bearbeitung fest definierte und sinnvolle Arbeitsabschnitte vorgegeben waren, konnte nach einem klaren "Gerüst" vorgegangen werden. Die Arbeit nach dem Vorgehensmodell erwies sich daher als gewinnbringend.

8 Abkürzungen

API	Application Programming Interface
ARP	Address Resolution Protocol
ASCII	American Standards Code for Information Interchange
AWT	Abstract Window Toolkit (Java)
BOOTP	BOOTstrap Protocol
DIN	Deutsches Institut fü Normung
DNS	Domain Name System
DRAM	Dynamic Random Access Memory
DTR	Data Terminal Ready
FRM	Frame
FTP	File Transfer Protocol
GIF	Graphics Interchange Format
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
ICA	Independent Computing Architecture
ID	Identification
IDE	Integrated Drive Electronics
IEEE	Institute of Electrical and Electronics Engineers
IMAP	Interactive Mail Access Protocol
I/O	Input/Output
IP	Internet Protocol
ISA	Industry-Standard Architecture
ISO	International Standards Organization
ISP	Internet Service Provider
JDK	Java Developers Kit
JIT	Just In Time (Java)
JNI	Java Native Interface
JPEG	Joint Photographic Experts Group
JVM	Java Virtual Machine
LAN	Local Area Network

LCD	Liquid Crytal Display
MAC	Media Access Control
NC	Network Computer
NCRP	Network Computer Reference Platform
NFS	Network File System
NS	Network Station
NSM	Network Station Manager
NTP	Network Time Protocol
NVRAM	Non-Volatile Random Access Memory
OLE	Object Linking and Embedding
OPC	OLE for Process Control
OS	Operating System
PCI	Peripheral Component Interconnect
PCMCIA	Personal Computer Memory Card International Association
POP	Post Office Protocol
POST	Power On Self Test
RAM	Random-Access Memory
RFC	Request For Comments (Internet)
ROM	Read-Only Memory
SGML	Standard Generalized Markup Language
SIMM	Single Inline Memory Module
SNMP	Simple Network Management Protocol
SRAM	Static Random Access Memory
SXGA	Super eXtended Graphics Adapter/Array
SVGA	Super Video Graphics Array
TCP/IP	Transmission Control Protocol/Internet Protocol
TFTP	Trivial File Transfer Protocol
URL	Uniform Resource Locator
VGA	Video Graphics Array/Accelerator
VRAM	Video Random Access Memory
WWW	World Wide Web

9 Glossar

Abstract Class Eine Kategorie, die nie instanziert sein sollte, nur ihre Unterklassen sollten instanziert werden. Abstrakte Klassen werden definiert, damit andere Klassen von ihnen den Namen übernehmen können.

Applet Ein Programm, das in Java geschrieben ist und innerhalb von HotJava und anderen Javafähigen Web-Browsern ausgeführt wird.

ASCII Eine Standardanweisung für Zeichen mit numerischem 7-bit Codes.

Bytecode Plattformunabhängiger Code, der durch den Java Compiler festgelegt und vom Java Interpreter ausgeführt wird.

Class Klassen enthalten Objekte und die zugehörigen Operationen auf diese.

Class Method Statische Methode für Zugriff auf statische Objektkomponenten. Sie tragen das Schlüsselwort „static“ und werden über den Namen der Klasse und den Punkt und Methode aufgerufen.

Class Variable Ein Datenelement, das mit einer bestimmten Klasse als vollständiges und nicht bestimmten Instanz der Klasse verbunden ist. Klassen Variablen werden in den Definitionen definiert.

Compiler Ein Programm zur Übersetzung von Quellcodes in den durch einen Computer auszuführenden Code. Der Java Compiler übersetzt den Java Quellcode in den Java Bytecode.

Constructor Eine Methode, durch die ein Objekt erstellt wird. In Java sind Konstruktoren Instanz Methoden mit dem gleichen Namen wie ihre Klasse. Java Konstruktoren werden mit dem Schlüsselwort “new“ hervorgerufen.

Exception Ein Signal, das meldet, daß irgendeine Ausnahmebedingung, wie beispielsweise ein Fehler, aufgetreten ist. Damit wird der normale Ablauf des Programms verhindert.

Garbage Collection Die automatische Abfrage und das Freigeben des Speichers, der nicht mehr in Gebrauch ist. Das Java Laufzeitsystem führt diese Garbage Collection durch, so daß der Programmierer nie explizit Objekte und andere Daten freigeben muß.

GUI Graphische Benutzerschnittstelle. Bezieht sich auf die Techniken, welche den Benutzern eine einfache grafische Schnittstelle mit Hilfe einer Tastatur und einer Maus zur Verfügung stellen.

Hierarchie Einteilung der Klassenverhältnisse in Java. Jedes Einzelteil kann ein oder mehrere Einzelteile in der Hierarchie unter sich haben. Die oberste Klasse wird als Wurzel bezeichnet. In der Java Klassen Hierarchie ist die Wurzel die Objekt Klasse.

HTML Datei Format, gegründet auf SGML, für Hypertext Dokumente auf dem Internet. Es ist sehr einfach und läßt das Einbetten von Bildern, Tönen, Videostreams, Formularfeldern und einfacher Textformatierung zu. Referenzen zu anderen Objekten werden mit URLs eingebettet.

HTTP Internet Protokoll, das verwendet wird, um Hypertext Objekte von dem fernen Host zu holen. Es basiert auf TCP/IP.

HUB Aktiver oder passiver Sternverteiler, der für Ethernet verwendet wird.

Instance Objekt einer bestimmten Klasse. In Java wird eine Instanz einer Klasse mit dem “new“ Operator, gefolgt vom Klassen Namen, erstellt.

Instance Method Eine beliebige Methode, die durch die Instanz einer Klasse hervorgerufen werden kann, aber den Klasse Namen nicht verwendet. Instanz Methoden werden in den Klassen Deklarationen definiert.

Instance Variable Ein beliebiges Einzelteil von Daten, das sich auf ein bestimmtes Objekt bezieht. Jede Instanz einer Klasse hat ihre eigene Kopie der Instanz Variablen, die in der Klasse definiert werden.

Interface Gruppe von Methoden in Java, die durch einige Klassen eingeführt werden können, unabhängig davon, wo die Klassen in der Klassen Hierarchie sind.

Internet Ein enormes Netz, das buchstäblich aus Millionen Hosts, vielen Organisationen und Ländern rund um die Welt besteht. Es wird physikalisch aus vielen kleineren Netzen zusammengefügt und durch einen allgemeinen Satz von Protokollen zusammengehalten.

Intranet Ein internes Unternehmensnetz, innerhalb dessen gleiche Schnittstellen und Protokolle einschließlich der Netzwerk Infrastruktur verwendet werden.

Interpreter Ein Modul, das wechselnd jede Anweisung in irgendeinem Körper des Codes decodiert und durchführt. Der Java Interpreter decodiert und führt Java Bytecode durch.

JPEG Ein Komprimierungsstandard für Bilddateien. Er erzielt eine enorme Komprimierung auf Kosten der Verzerrungen im Bild, die fast immer unerkennbar sind.

Linker Ein Modul, das ein ausführbares, komplettes Programm von den Teilmaschine Codemodulen aufbaut. Der Java Linker erstellt ein ablauffähiges Programm von kompilierten Klassen.

Local Variable Ein Datenelement, das innerhalb eines Blockes bekannt ist, aber außerhalb des Blockes nicht erreichbar ist. Beispielsweise ist die Variable, die innerhalb einer Java Methode definiert wird, eine lokale Variable und kann nicht außerhalb der Methode verwendet werden.

Method Eine Funktion, die in einer Klasse definiert wird.

Multithreaded Beschreibt ein Programm, das Teile seines Codes gleichzeitig durchführen kann.

Object Die Grundbausteine der objektorientierten Programme. Jedes Objekt ist eine Programmierungseinheit, die aus Daten (Instanz Variablen) und Funktionalität (Instanz Methoden) besteht.

Package Gruppe von Klassen in Java. Pakete werden mit dem Paketschlüsselwort erklärt.

Subclass Eine Klasse, die von einer bestimmten Klasse abgeleitet ist. Sie hat möglicherweise eine oder mehrere Klassen unter sich.

Superclass Eine Klasse, von der weitere Klassen abgeleitet werden. Möglicherweise beinhaltet sie auch noch Klassen.

Thread Die grundlegende Einheit des Programmablaufs. In einem Prozeß können einige Threads gleichzeitig laufen. Sie sind eigenständige Prozesse, die verschiedene Aufgaben erledigen können. Wenn ein Thread seinen Job beendet hat, wird er verschoben oder zerstört.

Unicode Ein 16-bit Zeichensatz definiert von ISO 10646

URL Ein Standard zum Schreiben einer Textreferenz auf einem willkürlichen Stück Daten in das WWW. Ein URL hat die Form "Protokoll://host/Localinfo", bei der „Protokoll“ ein Protokoll spezifiziert, um es dazu zu verwenden, die Nachricht zu holen (wie HTTP oder ftp). „Host“ spezifiziert den Internet-Namen des Hosts, auf dem „localinfo“ zu finden ist. Letzteres ist eine Zeichenkette (häufig ein Dateiname).

Variable Datenwort, das durch einen Bezeichner benannt wird. Jede Variable hat einen Typ wie integer oder Objekt und ein Gültigkeitsbereich.

10 Literatur

Bücher, Skripte und Zeitschriften

[1] **Auel Kersten:** *Not Completed, Hard- und Softwarehersteller entwickeln gemeinsamen NC, iX 5/97, S. 130*

[2] **Behme Henning:** *Duales System, Suns JavaStations 1 und 1+,* iX 4/97, S. 52

[3] **Flanagan, David**: *Java in a Nutshell, 2.Aufl.- Köln: O´Reilly, 1998*

[4] **Gordon, Rob**: *Essential Java Native Interface,Prentice Hall PTR, 1998*

[5] **Göhner, Peter:** *Skript zur Vorlesung Softwaretechnik I, IAS, Stuttgart, 1997/98*

[6] **Göhner, Peter:** *Skript zur Vorlesung Softwaretechnik II, IAS, Stuttgart, 1997*

[7] **IBM** *Network Computer Reference Platform Technical Specification: Release 1.0, 1997*

[8] [**BM** *Network Station Manager Installations- und Benutzerhandbuch: Juli 1998, SW NLS Center Kst. 2877 (IBM Form SC41-0664-01)*

[9] **JavaOS-index.html 1.13,** *27.10.97*

[10] **Lemay Laura, Perkins Charles L.:** *Java 1.1 in 21 Tagen, SAMS,1977*

[11] **Stelow Matthias:** *Terminal und mehr, Test vom IBMs Network Station ,iX 11/1997, S. 74 ff*

Online-Literaturen

[12] **ADS-tec GmbH:** *http://www.ads-tec.de/*

[13] **Bauer Homepage:** *accesses the serial ports , http://privat.swol.de/MarkusBauer/*

[14] **Central Data:** *portio Serial and Parallel Port Driver, http://www.cd.com/portio/*

[15] **Citrix Previews Java ICA:** *http://www.citrix.com/news/releases/maoct01.htm*

[16] **Corel: Office for Java:** *http://www.corel.com/news/1997/april/GONEpurejava.htm*

[17] **EOS, PortIO Overview:** *http://www.eos.dk/beans/portio/index.html*

[18] **Hugh Symons Group Plc:** *http://www.hughsym.co.uk*

[19] **IBM Network Computer:** *http://www.pc.ibm.com/networkstation/*

[20] **Java Communications API 1.1, 1.3 & 2.0:** *http://java.sun.com/products/javacomm/*

[21] **Java Developer Connection:** *http://java.sun.com/products/javaos/javaos.white.html*

[22] **Java Development Kit (JDK 1.1.4 & 6):** *http://java.sun.com/products/jdk/1.1.6/*

[23] **JavaOS: a Standalone Java Environment:**

http://java.sun.com/marketing/collateral/os.html

[24] **Java Technology Homepage:** *http://java.sun.com/products/javaos/javaos.white.html*

[25] **PC Support in Canada:** *http://www.can.ibm.com/helpware/networkstation.html*

[26] **Solutions Consulating, SerialPort, Java Serial Port Control:**

http://www.sc-systems.com/

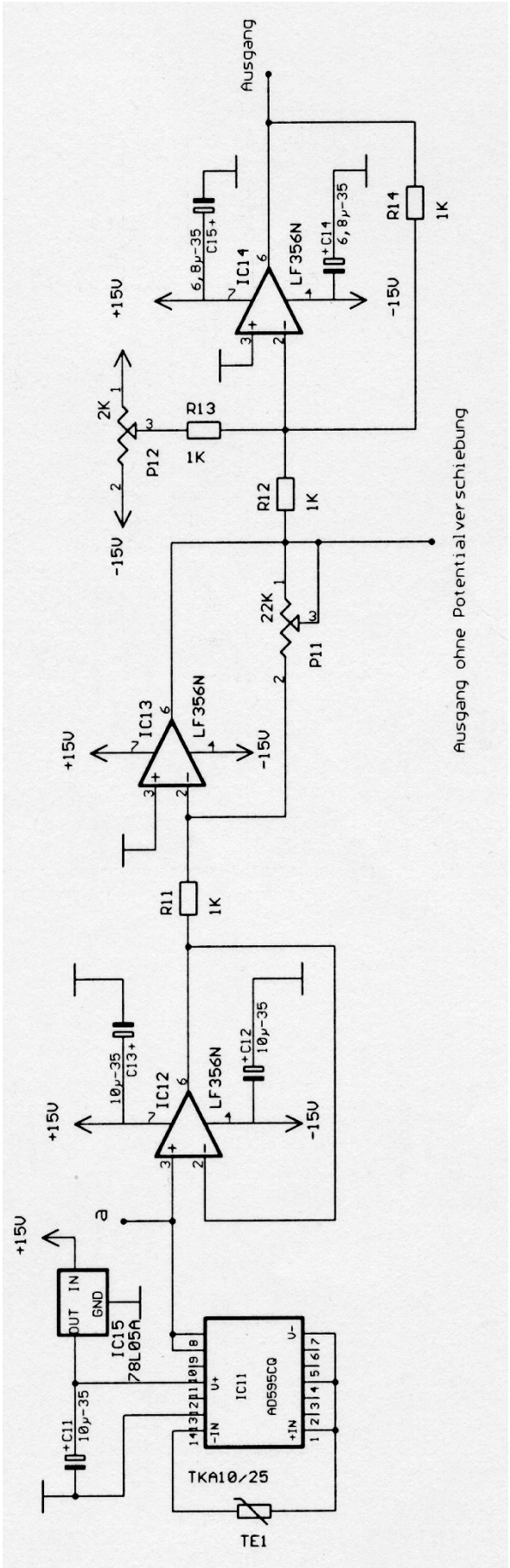
[27] **The Java Tutorial:** *http://java.sun.com/docs/books/tutorial/index.html*

[28] **Walusoft:** *http://www.walusoft.co.uk*

11 Anhang

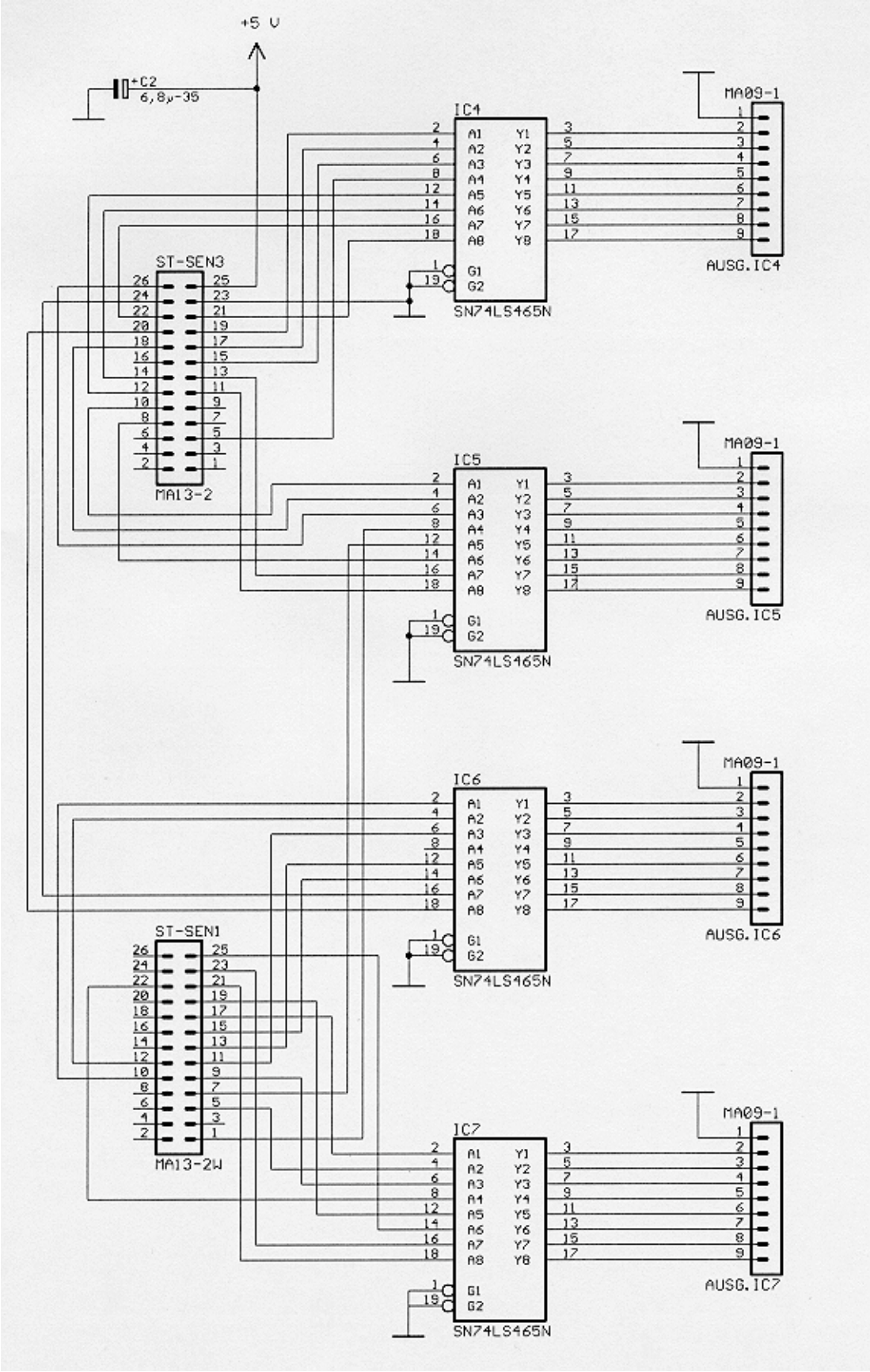
- Schaltplan des Temperaturverstärkers
- Schaltplan der Signalpuffern
- Schaltplan der Aktorumschaltern
- Netzteil und Spannungsregler
- Bestückungsplan der TPU*-Platine
- Bestückungsliste der TPU-Platine und des Netzteils
- Layout der TPU-Platine
- Pinbelegung der Temperaturschnittstelle
- Pinbelegung der Sensorik
- Pinbelegung der Aktorik
- Datenblatt Thermocoax Typ K
- Datenblatt AD595CQ
- Datenblatt LF356N
- Datenblatt SN74LS465N
- Datenblatt 74LS157PC

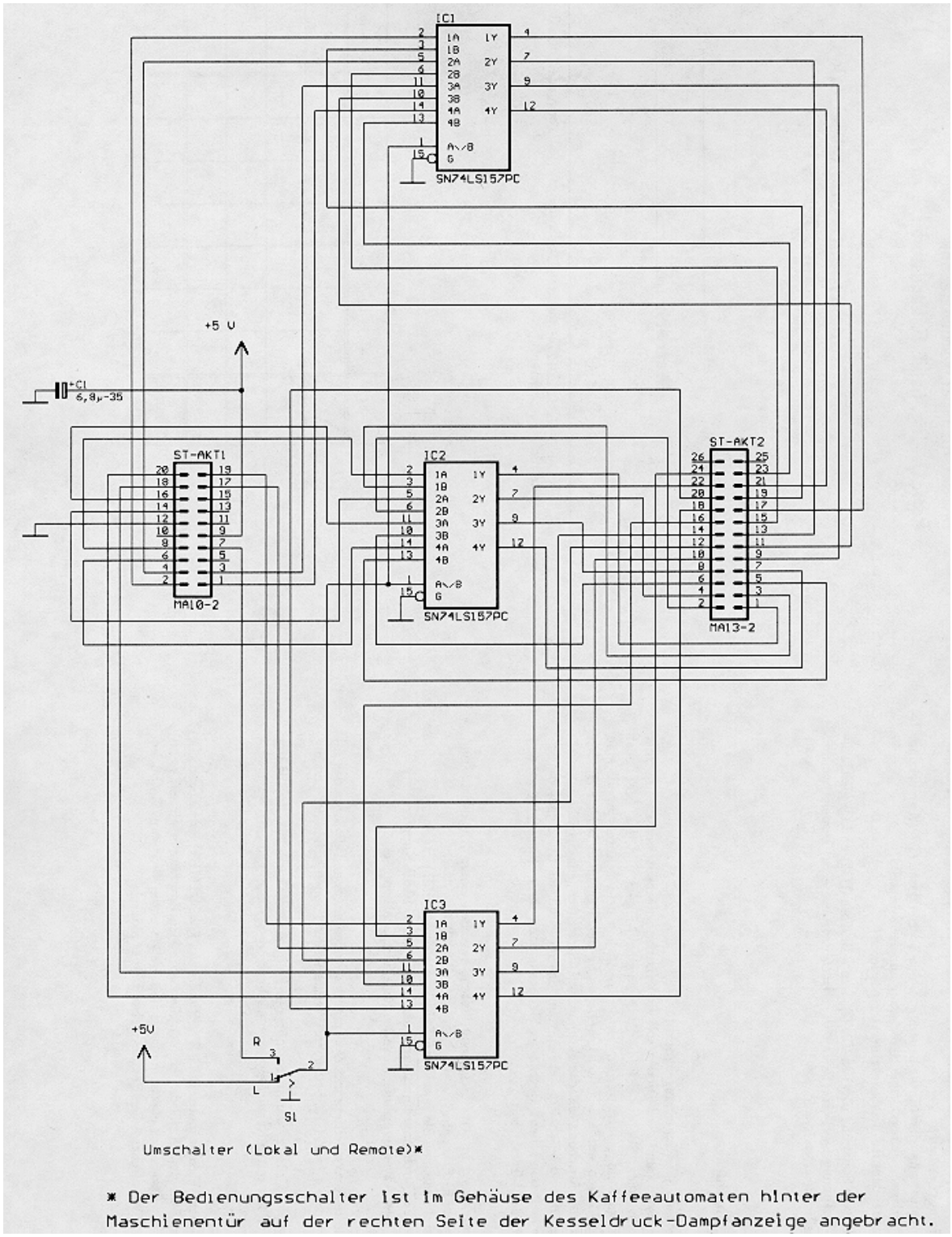
* Temperatur Puffer Umschalter

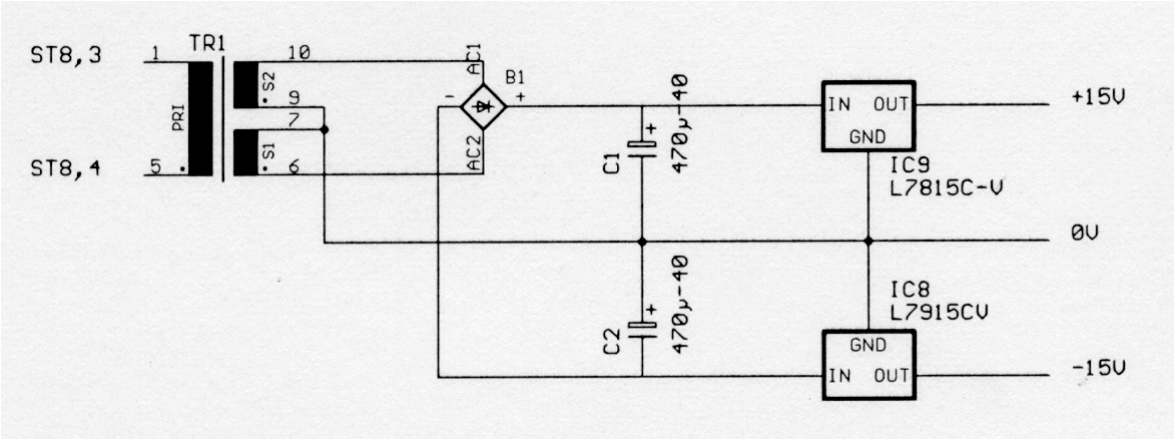


Es befinden sich vier gleichwertige Verstärkerschaltungen auf der TPU-Platine, deren Ausgänge auf die zugehörigen Schnittstelle (Temp auf der Rückseite) geführt sind. Jeder Verstärker ist mit einer Buchstaben versehen. Diese sind wie folgt eingeordnet:

A Brühwasser
B Trichter
C Raumtemperatur
D frei als Reserve


+5 V
+C2
6,8µ-35
IC4
A1 Y1
A2 Y2
A3 Y3
A4 Y4
A5 Y5
A6 Y6
A7 Y7
A8 Y8
G1
G2
SN74LS465N
MA09-1
AUSG.IC4
ST-SEN3
MA13-2
IC5
AUSG.IC5
IC6
AUSG.IC6
ST-SEN1
MA13-2W
IC7
AUSG.IC7


IC1
SN74LS157PC
+5 V
+C1
6,8µ-35
ST-AKT1
MA10-2
IC2
SN74LS157PC
ST-AKT2
MA13-2
IC3
SN74LS157PC
+5V
R
L
S1
Umschalter (Lokal und Remote)*
* Der Bedienungsschalter ist im Gehäuse des Kaffeeautomaten hinter der Maschienentür auf der rechten Seite der Kesseldruck-Dampfanzeige angebracht.



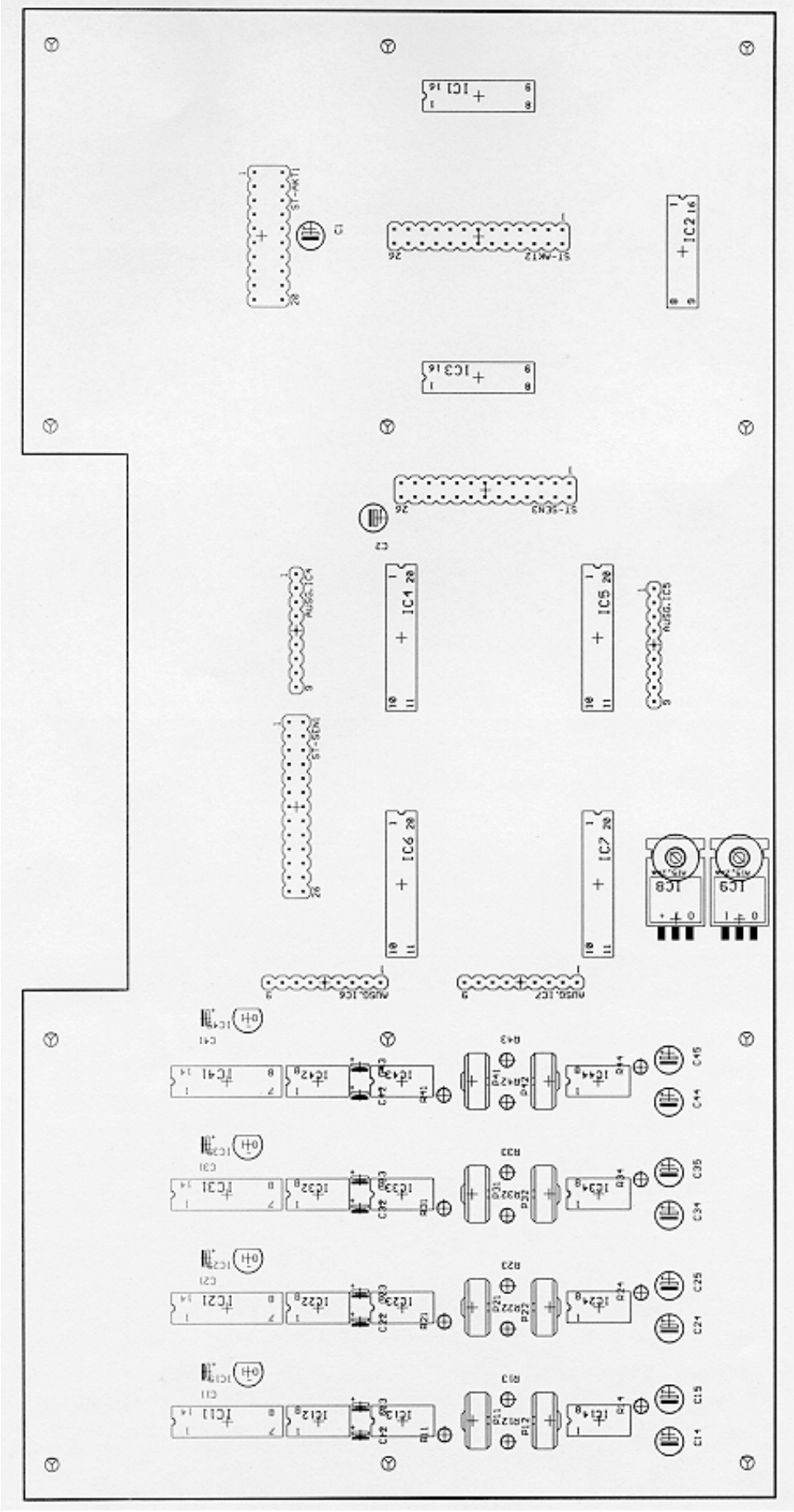
Bemerkungen:

Die Spannungsregler IC8 und IC9 sind auf der TPU-Platine zu finden.

Der ST8 (Stecker) gehört zu der Elektronikkarte des Kaffeeautomaten.

Die Elektronikkarte versorgt die Umschalter und Signalpuffer mit +5V.

Für eine konstante Versorgungsspannung des AD595CQ sind weitere +5V Regler auf der TPU-Platine vorgesehen.



Bestückungsliste der TPU-Platine

IC1	SN74LS157PC
IC2	SN74LS157PC
IC3	SN74LS157PC
IC4	SN74LS465N
IC5	SN74LS465N
IC6	SN74LS465N
IC7	SN74LS465N
IC8	L7915CV
IC9	L7815C-V
C1	6,8µ-35
C2	6,8µ-35

Verstärker A	Verstärker B	Verstärker C	Verstärker D	
IC11	IC21	IC31	IC41	AD595CQ
IC12	IC22	IC31	IC42	LF356N
IC13	IC23	IC33	IC43	LF356N
IC14	IC24	IC34	IC44	LF356N
IC15	IC25	IC35	IC45	78L05A
C11	C21	C31	C41	10µ-35
C12	C22	C32	C42	10µ-35
C13	C23	C33	C43	10µ-35
C14	C23	C34	C44	6,8µ-35
C15	C25	C35	C45	6,8µ-35
R11*	R21	R31	R41	1KOhm
R12*	R22	R32	R42	1KOhm
R13*	R23	33	R43	1KOhm
R14*	R24	R34	R44	1KOhm
P11*	P21	P31	P41	2KOhm
P12*	P32	P32	P42	22Kohm

*Alle R sind Metallwiderstände
*Alle Potentiometer sind Kermitbauteile

ST-AKT2	Pfostenstecker 20P
ST-SEN1	Pfostenstecker 26P
ST-SEN3	Pfostenstecker 26P

Bestückungsliste des Netzteils

TR1	Trafo, Typ EI42/14,8, 5VA, 2*15
B1	Gleichrichter (Brücke), KS 112A3880, C 1500/ 1000
C1	470µ-40
C2	470µ-40

Schalter am Kaffeeautomat

S1	Miniatur Griffschalter

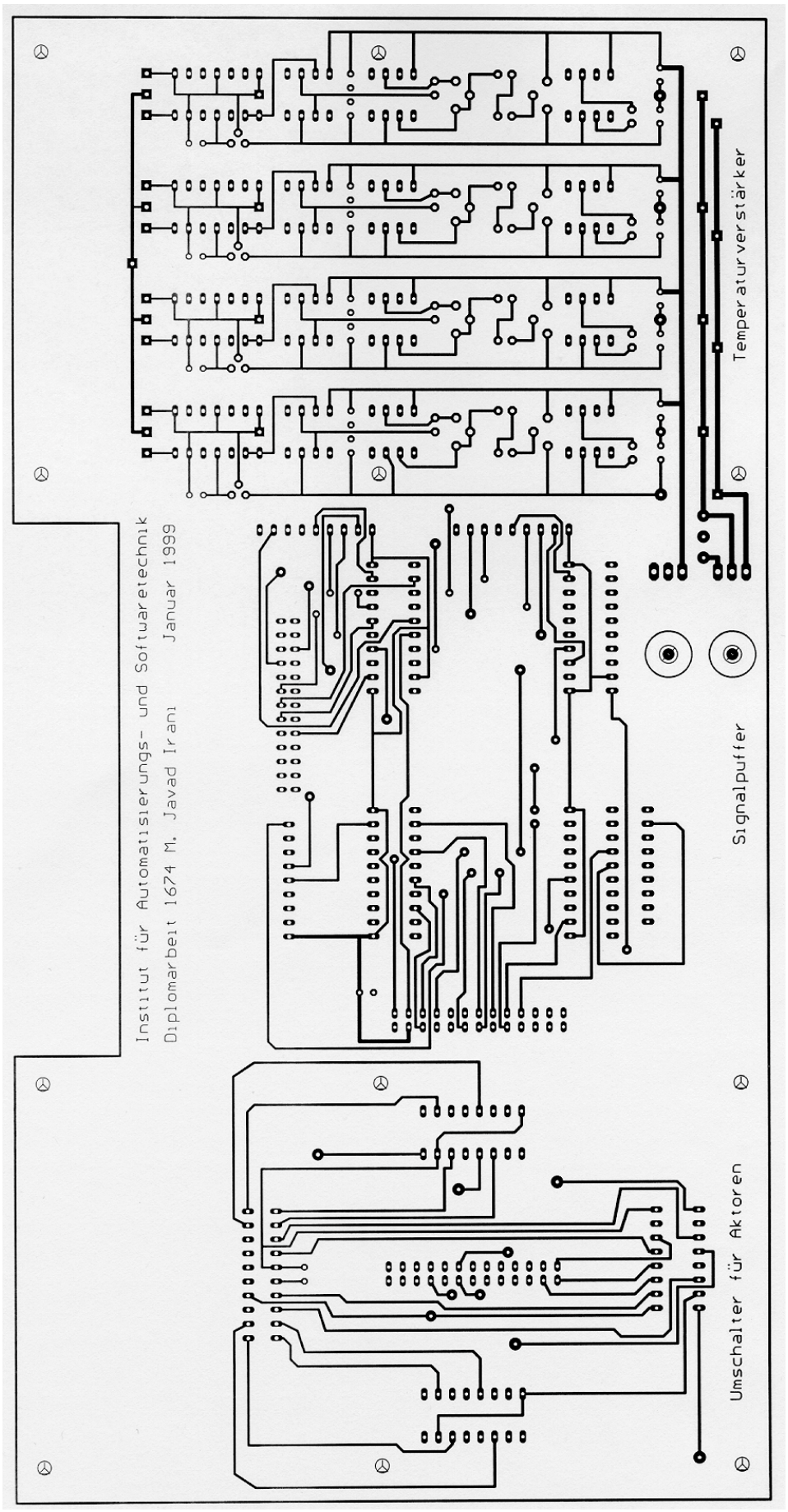
Schnittstellen

Temp	D-SUB 9 Buchse
Aktoren	D-SUB 15 Buchse
Sensoren	D-SUB 37 Buchse

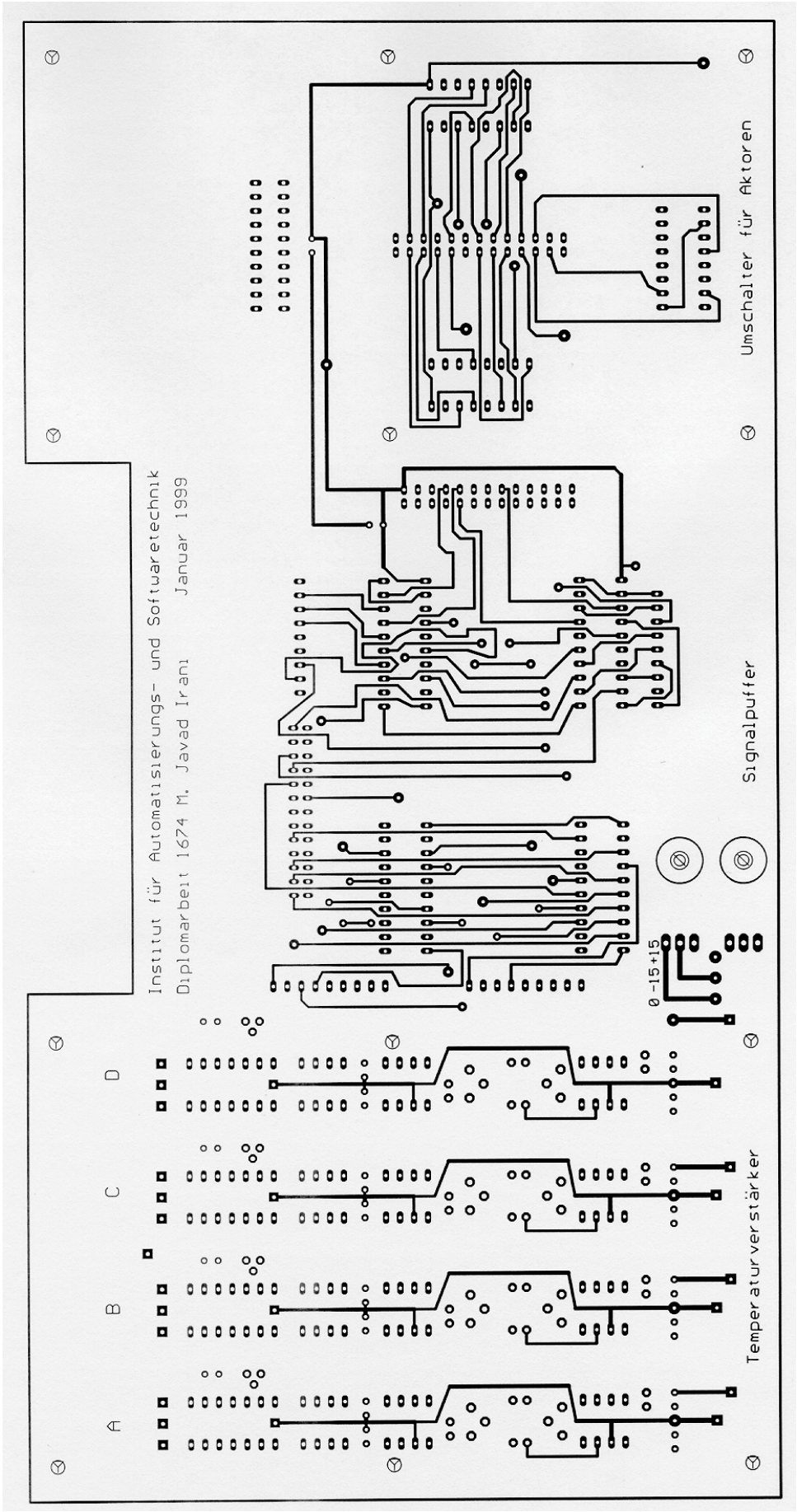
weitere Bauteile

Verbindungsstücke für Signalabzweigung zwischen der Elektronikkarte, der Treiberkarte und der TPU-Platine. Alle Verbindungsstücke sind Pfostenstecker und –buchse.

Thermoelement (Thermocoax) mit dem Type TKA 10/25 Typ K	3 Stücke


Temperaturverstärker
Signalpuffer
Umschalter für Aktoren
Institut für Automatisierungs- und Softwaretechnik
Diplomarbeit 1674 M. Javad Irani Januar 1999

Lötseite



Bestückungsseite

Pinbelegung der Temperaturschnittstelle

D-SUB, 9 Büchse (Kaffeeautomat) und D-SUB, 37 Büchse (PC; PCL-818L)

Pin-Nummer	Beschreibung	D-SUB, 37 Büchse
1	Brühwasser	1
2	Trichter	2
3	Raumtemperatur	3
4	frei	4
5	Masse	9
6-9	nicht belegt	5, 6, 7, 8 und 10-37 nicht belegt

Pinbelegung der Sensorik

D-SUB, 37 Büchse (Kaffeeautomat) und D-SUB, 37 Büchse (PC; PET48DIO)

Pin-Nummer	Beschreibung	CN1	CN2
1	Heißwasserventil	1	
2	Speiseventil	3	
3	Dampfventil	5	
4	Heizung-Dampfkessel	7	
5	Brühventil	9	
6	Einspülventil	11	
7	Heizung-Boiler	13	
8	Reinigungsventil	15	
9	Schalter 1	17	
10	Brühmotor-rückwärts	19	
11	Espressopumpe	21	
12	Dampfdruckregler 2	23	
13	Dampfdruckregler 1	25	
14	Schalter 2	27	
15	Handeinwurf	29	
16	Brühmotor-vorwärts	31	
17	nicht belegt		
18	nicht belegt		
19	Masse	38	38
20	KMP 1-leer		1
21	KMP-Taste		3
22	KMP-Mühle		5
23	KMP 2		7
24	DFM (Durchflußmengenmesser)		9
25	Reset		11
26	Bestück-Signal		13
27	Reserve		15
28	Wasserdruck		17
29	END 1, (Brüher-Endschalter)		19
30	END 2		21
31	END 3		23
32	KMP 1-Impulse		25
33	Kaffeemehlportionierer (KMP) 2-leer		27
34	KMP 2-Impulse		29
35	nicht belegt		
36	nicht belegt		
37	nicht belegt		

Pinbelegung der Aktorik

D-SUB, 15 Büchse (Kaffeeautomat) und D-SUB, 15 Büchse (PC; PCL-818L)

Pin-Nummer	Beschreibung	CN1, DIGO
1	Tastensteuerung für Lokal und Remote	1
2	Mengen-Vorwahltaste „+“	3
3	Mengen-Vorwahltaste „-„	5
4	Dampfausgabetaste	7
5	Espresso-Taste „2 Tassen“	9
6	Espresso-Taste „1 Tassen“	11
7	Cafe´ creme´ „2 Tasse“ (Schümli)	13
8	Masse	17
9	Cafe´ creme´ „1 Tasse“ (Schümli)	2
10	Spezialtaste	4
11	Filterkaffee-Taste „Kännchen“	6
12	Filterkaffee-Taste „Tasse“	8
13	Heißwasserausgabe-Taste „große Menge“	10
14	Heißwasserausgabe-Taste „kleine Menge“	12
15	nicht belegt	14-16, 18-20 nicht belegt